REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
June 13, 1997

3. REPORT TYPE AND DATES COVERED
Final Technical, August 1993-Feb. 1997

4. TITLE AND SUBTITLE
A Unified Framework for Verification and Complexity
Analysis of Real-Time and Distributed Systems

5. FUNDING NUMBERS
F49620-94-1-0199

AFOSR-TR-
97-0648

6. AUTHOR(S)
Nancy Lynch

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
(Sponsoring Scientific Office)
Airforce/NM
110 Duncan Avenue B115
Bolling AFB, DC 20332-8080

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

N/A

12a. DISTRIBUTION/AVAILABILITY STATEMENT

No limits of disclosure

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

We have developed the *timed I/O automaton* model, a basic compositional formal model for describing and analyzing real-time systems and distributed systems (in particular, distributed systems with precise timing assumptions and requirements). We have developed proof techniques, both manual and computer-assisted, for use with timed I/O automata, and have used the model and methods for analyzing a variety of problems and systems. These examples arise from a diverse set of application areas, including connection management protocols, clock synchronization, fault-tolerant distributed consensus, group communication, and real-time process control systems.

We have extended the basic timed I/O automaton model in three directions: to include liveness constraints (*live timed I/O automata*), hybrid continuous/discrete behavior (*hybrid I/O automata*), and probabilistic behavior (*probabilistic timed I/O automata*). In each case, we have developed proof methods and have applied the models and methods to substantial problems. For example, in the hybrid systems area, we have carried out an extended case study of safety aspects of automated transportation systems.

We have recently begun the development of a programming language/environment, based upon our formal models, and intended to support the coordinated development and analysis of distributed systems.

14. SUBJECT TERMS

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to *stay within the lines* to meet *optical scanning requirements*.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C -	Contract	PR -	Project
G -	Grant	TA -	Task
PE -	Program Element	WU -	Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

DOD -	See DoDD 5230.24, "Distribution Statements on Technical Documents."
DOE -	See authorities.
NASA -	See Handbook NHB 2200.2.
NTIS -	Leave blank.

Block 12b. Distribution Code.

DOD -	Leave blank.
DOE -	Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
NASA -	Leave blank..
NTIS -	Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

Contents

Final Technical Report for AFOSRF49620-94-1-0199
A Unified Framework for Verification and Complexity Analysis of Real-Time and Distributed Systems

Period of Grant: August 1993-February 1997

Principal Investigator: Prof. Nancy Lynch,
lynch@theory.lcs.mit.edu, (617)253-7225

June 16, 1997

Research Staff:

Nancy Lynch, TDS Group Leader
Nir Shavit, Visiting Professor, Tel Aviv University
Alex Shvartsman, Research Associate
John Lygeros, Research Associate

Visitors:

Vicente Cholvi-Juan, Unitat Predepartamental d'Informatica
Alan Fekete, University of Sydney
Gunter Leeb, University of Austria
Sergio Rajsbaum, Ciudad Universitaria
Dan Touitou, Tel Aviv University
Frits Vaandraager, University of Nijmegen
Asaph Zemach, Tel Aviv University

Graduate Students:

Sudhanshu Aggarwal, completed MS
Oleg Cheiner
Gio Della-Libera, completed MS
Roberto De Prisco, completed MS
Rainer Gawlick, completed PhD
Gunnar Hoest
Henrik Jensen
Roger Khazan
Carl Livadas
Victor Luchangco, completed MS
Boaz Patt-Shamir, completed PhD
Anna Pogosyants
Isaac Saias, completed PhD
Roberto Segala, completed MS, PhD
Mark Smith, completed MS
Jørgen Søgaard-Andersen, completed PhD
Ekrem Söylemez, completed MS
Mandana Vaziri, completed MS
H. B. Weinberg, completed MS

Undergraduate Students:

Ekaterina Dolginova
Tsvetomir Petrov

Abstract

We have developed the *timed I/O automaton* model, a basic compositional formal model for describing and analyzing real-time systems and distributed systems (in particular, distributed systems with precise timing assumptions and requirements). We have developed proof techniques, both manual and computer-assisted, for use with timed I/O automata, and have used the model and methods for analyzing a variety of problems and systems. These examples arise from a diverse set of application areas, including connection management protocols, clock synchronization, fault-tolerant distributed consensus, group communication, and real-time process control systems.

We have extended the basic timed I/O automaton model in three directions: to include liveness constraints (*live timed I/O automata*), hybrid continuous/discrete behavior (*hybrid I/O automata*), and probabilistic behavior (*probabilistic timed I/O automata*). In each case, we have developed proof methods and have applied the models and methods to substantial problems. For example, in the hybrid systems area, we have carried out an extended case study of safety aspects of automated transportation systems.

We have recently begun the development of a programming language/environment, based upon our formal models, and intended to support the coordinated development and analysis of distributed systems.

1 Introduction

Our AFOSR project entitled "A Unified Framework for Verification and Complexity Analysis of Real-Time and Distributed Systems" began in August, 1993 and continued through February, 1997. Its stated purpose was to develop a general formal semantic model for reasoning about real-time and distributed systems, and to establish its value by using it in several ways: for proving fundamental results about the capabilities of real-time and distributed systems, for the description of practical systems, for the specification of interesting problems to be solved in real-time and distributed systems, and for analysis of system performance. The "grand vision" behind this work was (and is) the eventual production of a coordinated suite of practical development tools and practical verification/analysis tools for real-time and distributed systems, all based firmly on a good mathematical foundation. This work built upon our prior work on developing models for untimed systems, in particular, the I/O automaton model of Lynch and Tuttle [81] and the untimed automaton model of Lynch and Vaandrager [76].

During the 3 1/2 years of this contract, we reached most of our goals. In this Introduction, we describe some of the highlights of the project. More information about the specific accomplishments appears in the following sections. The individual items in those sections include URL pointers that yield additional information about the individual accomplishments. Our group's entire Web site begins at URL `http://theory.lcs.mit.edu/tds`.

1.1 Models and Proof Methods

We completed most of the work on the "core" model, which we call the *timed I/O automaton model* (or just the *timed automaton* model), during the first year of the contract. Besides defining the model, we formulated compositional, invariant assertion, simulation relation and temporal logic proof methods in terms of the model, as well as a significant body of process algebraic methods. The model is capable of expressing safety and real-time (e.g., performance) properties, as well as some liveness properties. The resulting model is an improved version of earlier models by Lynch and Vaandrager, improved by addition of such features as incremental time, explicit trajectories, and components with local clocks that progress at different rates. The model is also an extension of our earlier models for untimed systems.

An interesting feature of this work is our use of invariant assertion and simulation relation techniques to prove timing properties – this is an advance over their common use to prove "ordinary" safety properties. In joint work with Garland and Guttag, we automated the invariant assertion and simulation relation techniques using the Larch Prover. Archer and Heitmeyer at the Naval Research Laboratory also automated these methods using PVS.

We used the model and methods for analyzing a variety of problems and systems, arising from a diverse set of application areas. The point of these cases studies was twofold: they were intended to contribute useful results in their respective application areas, and they were intended to assist us in the development and validation of our model. The case studies were chosen from many areas, including connection management protocols, clock synchronization, fault-tolerant distributed consensus, group communication, and real-time process control systems.

Motivated by some of the applications we considered, we extended the basic timed automaton model in three different directions: to include general liveness constraints (*live timed I/O automata*), hybrid continuous/discrete behavior (*hybrid I/O automata*), and probabilistic behavior (*probabilistic timed I/O automata*). Each of these three developments was itself a substantial modelling effort. Each model is compositional, and supports its own suite of proof methods. The work on the general liveness model represents a major simplification over previous attempts at compositional liveness. The hybrid system model is very general, and supports composition using continuously-changing shared variables as well as shared actions. The probabilistic model is the first formal model that is powerful enough to permit accurate description and analysis of realistic randomized distributed algorithms. For now, these three different extensions are separate; we have not yet integrated the complications of liveness, hybrid behavior, and probabilities into one coherent model.

We also applied these models to substantial case studies. For the liveness model, the main applications were to connection management protocols. For the hybrid system model, the main application was to safety aspects of automated transportation systems. The probabilistic model was applied to the task of analyzing and verifying randomized distributed algorithms arising in the PODC community.

1.2 Applications

Our first major case study involved communication systems. The first of these, carried out jointly with Lampson, was a study of the five-packet interchange protocol of Belsnes and of a timing-based protocol of Liskov, Shrira and Wroclawski. Our work demonstrated how both of these protocols could be viewed formally as implementations of a common generic protocol. A continuation of this project (still being completed) involved modelling and analysis of TCP and T/TCP; this work involved collaboration with Clark. It produced not only models and correctness proofs for these protocols, but also an impossibility result expressing an important limitation on their behavior. Our communication work was based on a mixture of timed and untimed automata, and included safety, liveness and timing properties. It led to our establishment of formal embeddings of untimed models within timed models, which are needed to allow timing-based algorithms to implement untimed specifications.

We also carried out an extended case study in the area of real-time process control, for automated transportation systems. Working with system developers at Raytheon and the California PATH project, and building on our hybrid automaton model, we have developed models for some mechanisms that ensure safety in certain automated transportation systems. This work helped us to develop our hybrid automaton model. It also led to substantive information (careful safety guarantees, limitations on capabilities) about both automated transportation systems we have studied.

A third important case study involved the definition and analysis of *building blocks* for the construction of efficient, fault-tolerant distributed systems. This work included an analysis of the key components of the Orca distributed shared memory system. We also developed a notion of *eventually serializable data service*, which models certain weakly coherent data services that are important in practice. We developed a new, simple specification for a *view-synchronous group communication* service, and showed how it can be used to implement a totally-ordered (non-view-oriented) group communication service. Other work in this area included a combined broadcast-convergecast communication primitive, and the development of a practical fault-tolerant distributed consensus protocol. In this work, timed automata formed the basis for practical time performance analysis.

Other case studies involved probablistic systems.

1.3 Algorithms

We obtained new algorithms and impossibility results in the areas of communication protocols, asynchronous computability, clock synchronization, and self-stabilizing systems. We (primarily, Shavit and graduate students) developed a collection of highly efficient concurrent data structures for use in shared-memory multiprocessors and local area networks. We developed a neat formal decomposition and proof for the Borowsky-Gafni fault-tolerant simulation algorithm.

Lynch wrote an 800-page graduate textbook presenting the most important results of the research area of Distributed Algorithms, unified in terms of our untimed and timed automaton models. This book is currently being used as a graduate text in several institutions. Also, Shvartsman wrote a book on fault-tolerant parallel computing.

1.4 Tools

We are completing a preliminary design of a programming language, IOA, for I/O automata, intended to support the coordinated development and analysis of distributed systems. We intend for the IOA language to be integrated with a variety of tools, including simulators, theorem-provers, and model-checkers. Eventually, it should permit generation of working distributed code.

Our preliminary design does not yet include timing features – this first cut is based just on our untimed model.

2 Models and proof methods

In this section, we describe our specific projects on development of formal models and accompanying proof and analysis methods.

2.1 The timed automaton model

During the first year of the contract, we completed most of the work on the "core" timed automaton model. Besides defining the model, we formulated compositional, invariant assertion, simulation relation and temporal logic proof methods in terms of the model, as well as a significant body of process algebraic methods. The model is capable of expressing safety and real-time (e.g., performance) properties, as well as some liveness properties.

The resulting model is an improved version of earlier models by Lynch and Vaandrager [70, 71, 72, 114] improved by addition of such features as incremental time, explicit trajectories describing state changes over continuous time, and components with local clocks that progress at different rates. The conversion from absolute to incremental time was an especially significant improvement in the model, because it simplified many of the definitions and proofs. The latest versions of our work appear in [74, 76, 77, 78, 82, 73, 75]. The model is also described in Lynch's book [58].
See URL `http://theory.lcs.mit.edu/tds/timed-aut.html`.
The temporal logic work appears in [112].

2.2 Using invariants and simulation relations to prove timing properties

We developed and exploited a method, suggested earlier by Lynch and Attiya, [80], for proving timing properties for timing-based systems. This method involves expressing the systems as timed automata, encoding time deadline information into the states of the automata, and including this time deadline information in invariants and simulation relations. For example, if we want to show that a timing-based system S meets a timing specification P, we express both S and P as timed automata, and demonstrate a formal simulation relationship between S and P. In many cases, the simulation relationship has an interesting form: a set of inequalities. The method involves proving that the inequalities hold initially and that they are preserved by all steps of the system. This method is developed and used for small-to-medium examples (a counter, Fischer's mutual exclusion protocol, simple communication protocols) in [54, 53, 57, 50, 51, 58, 113]. It is used for a much larger example, a timing-based connection management protocol, in [44, 111, 112].
See URL `http://theory.lcs.mit.edu/tds/FORTE93.html` and
URL `http://theory.lcs.mit.edu/tds/TR-589.html`.
It has also proved to be important in the hybrid systems work, discussed in Sections 2.6 and 3.2.

2.3 Mechanical verification

Garland, Luchangco, Lynch and Söylemez [113, 50, 51] developed a method for computer-aided verification of timing properties of real-time systems. Namely, a special case of the general timed automaton model, along with the invariant assertion and simulation techniques described in Section 2.2, were formalized within the Larch Shared Language. The Larch Prover was then used to carry out formal proofs for two examples – a simple counter and Fischer's mutual exclusion protocol. This effort involved building a substantial amount of specialized machinery to enable the Larch Prover to manipulate our timed automaton model.
See URL http://theory.lcs.mit.edu/~victor_l/papers/FORTE94.html
and http://theory.lcs.mit.edu/~victor_l/papers/masters.html.

In other work on mechanical verification, Petrov, Pogosyants, Luchangco, Garland, and Lynch [86] developed a formal representation and computer-checked proof of correctness for the Dolev-Shavit Bounded Concurrent Timestamp algorithm [20], again using the Larch Prover. This algorithm is one of the most complicated in the distributed computing theory literature. Its proof uses invariant assertions and a simulation relation to a corresponding Unbounded Concurrent Timestamp algorithm, following a strategy developed earlier by Gawlick, Lynch, and Shavit [31, 28]. This work demonstrates that our methods work well for complex examples.
See URL http://theory.lcs.mit.edu/tds/CTSS.html.

Segala and Pogosyants carried out a computer-assisted proof of time performance for a randomized distributed algorithm, using the Larch Prover [87]. This work is based on our probabilistic model and proof methods, discussed in Section 2.7.
See URL http://theory.lcs.mit.edu/~segala/PS95.html.

Our work on mechanical verification of timed and untimed systems has led to various additions and improvements in the Larch system.

In related work on mechanical verification, Archer and Heitmeyer at the Naval Research Laboratory have used many of our proofs (e.g., those described in Sections 2.2, 3.2.1, 3.2.2, 3.2.3, and 3.3.4) as examples for their work on mechanical theorem-proving using PVS. We assisted by providing information as needed.

Our projects on computer-aided verification are described at:
URL http://theory.lcs.mit.edu/tds/cav.html.

2.4 Practical performance and fault-tolerance analysis

Building on earlier work of Patt-Shamir [84], De Prisco developed a new "Clock Timed Automaton" model [90]. This is a special case of a general timed automaton that includes an explicit notion of a

clock. It provides a systematic way of describing timing-based systems in which there is a notion of "normal" timing behavior, but that do not necessarily always exhibit this "normal" behavior. This model is intended to be used for stating and proving performance and fault-tolerance properties for practical systems. In particular, it is useful for properties that hold when the system stabilizes to a situation in which timing behavior is normal and no additional failures occur.
See URL `http://theory.lcs.mit.edu/tds/paxos.html`

Recently, in the course of our work on *view-synchronous group communication services* [26], we developed a new method for modular performance and fault-tolerance analysis. Like the method based on Clock Timed Automata, this method involves proving properties under certain "stabilized" conditions. However, the new method is described in a completely modular way, i.e., it allows proof of performance and fault-tolerance properties for a complex system by using such properties for the system's components.
See URL `http://theory.lcs.mit.edu/tds/vsgc.html`.

2.5 Liveness and timed automata

Segala, Gawlick, Søgaard-Andersen, and Lynch have incorporated general liveness properties into the timed automaton model, yielding a compositional model for general liveness properties [33, 32, 96]. An important aspect of these definitions is the ability to identify which liveness properties are guaranteeable by a system, no matter what its environment does; a key definition is that of a *receptive* live timed automaton. The main result is a compositionality result for such automata. The paper treats liveness properties for untimed automata as well as timed automata.
See URL `http://theory.lcs.mit.edu/tds/liveness.html`.

2.6 Hybrid automata

Hybrid systems are systems that exhibit both discrete and continuous behavior, for example, a process control system with a controller that is a distributed algorithm. Lynch, Segala, Vaandrager and Weinberg developed the *hybrid I/O automaton* model, a mathematical model based on labelled transition systems, designed for modelling and reasoning about hybrid (continuous/discrete) systems [67, 66]. The model includes trajectories, and continuous interaction among components. The model also includes composition and hiding operations, plus a notion of simulation relation to support reasoning using levels of abstraction. Finally, it includes a notion of receptiveness, which captures the idea that a hybrid automaton allows time to pass without bound.

The model supports composition, invariant assertion, and simulation relation proof techniques, based on a collection of theorems we have proved. The theorems showing how to prove invariants and simulations are especially nice, because they break down the facts to be proved into two completely separate categories: continuous facts and discrete facts. Continuous facts can be proved using methods of continuous mathematics (e.g., differential equations), while discrete facts can be proved using discrete math (e.g., algebraic deduction).

This model was presented at the 1995 DIMACS Workshop on Hybrid Systems, and a full journal version is still in progress. The full version will contain some technical generalizations of the original model.
See URL `http://theory.lcs.mit.edu/tds/hybrid-model.html`.

Branicky, Dolginova and Lynch [16] extracted some techniques involving reasoning about derivatives from the control-theory literature, and presented them in a form that is suitable for reasoning about hybrid systems modelled in terms of hybrid automata.
See URL `http://theory.lcs.mit.edu/tds/platoons.html`.

2.7 Probabilistic automata

Segala, with some collaboration with Saias, Pogosyants, and Lynch, developed a new and comprehensive formal model for randomized distributed systems, both untimed and timed, together with a toolkit of proof techniques for proving correctness and time performance properties of such systems [94, 93, 97, 98, 62, 87]. Segala's model was influenced by some ideas from the slightly earlier thesis of Saias [91]. These techniques include: rules for proving probabilistic time performance properties (progress statements) by combining probabilistic progress claims [62], rules for deriving expected time bounds from progress statements [62, 87], rules for composing probabilistic systems [94], rules for building such systems hierarchically [97, 98, 93], and rules (*coin lemmas*) for reducing probabilistic systems to non-probabilistic systems [62].
See URL `http://theory.lcs.mit.edu/tds/probability.html`.

Various formal techniques for proving probabilistic time performance properties for probabilistic systems are summarized in [56]. We use these techniques in a variety of case studies, described in Section 3.4.

Pogosyants, Segala and Lynch developed new coin lemmas for random walks, developed new general techniques for compositional reasoning about randomized distributed algorithms, and also developed some new modular techniques for time performance analysis. This work was carried out as part of a project [88, 89] on modelling and analyzing the very complex randomized consensus protocol of Aspnes and Herlihy [6]. This case study is discussed further in Section 3.4.
See URL `http://theory.lcs.mit.edu/tds/AH.html`.

2.8 Other work

Various other projects involving formal modelling were carried out in our group during the time of the contract. We list them briefly here. None of these dealt specifically with timing issues.

Segala published a paper based on his MS thesis, giving a process algebra for I/O automata [83]. That paper proposes process algebraic methods for proving properties of systems described as I/O automata. Lynch and Segala carried out a comparative study, applying both simulation techniques and process algebraic techniques to a simple concurrent system verification problem [63, 64, 65]. Other work by Segala dealt with notions of fairness and implementation [92, 95].

Jensen and Vaziri examined and developed techniques for the integration of model checking and theorem proving methods for verification of concurrent systems. Specifically, they studied the feasibility of abstracting from an infinite-state system to a finite-state system. They developed a property-preserving abstraction theorem for an untimed automaton model. They examined uses of this theorem in the verification of concurrent read/write and mutual exclusion algorithms.
See URL `http://theory.lcs.mit.edu/~ejersbo/research.html#absio.html`.

3 Applications

In this section, we describe our application case studies. We group them into four main areas: communication, real-time systems, distributed system building blocks, and probabilistic systems.

3.1 Communication

3.1.1 At-most-once message delivery protocols

Our first communication case study, which was completed during the beginning months of this contract, was an extensive study of the five-packet interchange protocol of Belsnes [14] and of a timing-based protocol of Liskov, Shrira and Wroclawski [48]. Our work demonstrated how both of these protocols could be viewed formally as implementations of a common generic at-most-once message delivery protocol [44, 111, 112]. This work was based on a mixture of timed and untimed automata, and included safety, liveness and timing properties. Proof techniques included invariants, simulation relations (both forward and backward), and temporal logic. Also, the mixture of timed and untimed models required us to define formal embeddings of untimed models within timed models; this was needed for claiming that timing-based systems implemented untimed specifications.
See URL `http://theory.lcs.mit.edu/tds/FORTE93.html`
and URL `http://theory.lcs.mit.edu/tds/TR-589.html`.

3.1.2 Connection management protocols

In a continuation (still being completed) of the project described in Section 3.1.1 [110], Smith worked with Lynch and Clark on modelling and analyzing the TCP and T/TCP Internet transport-level protocols. T/TCP is a new version of TCP, by Braden and Clark, that is more efficient for transactions (simple request-response pairs of messages). First, an abstract formal specification was developed for the user-visible behavior of TCP, and a formal proof (currently being polished) was developed for the fact that TCP satisfies this specification [109].

Next, an attempt was made to verify the correctness of T/TCP by means of a simulation relation mapping it to TCP. However, this attempt resulted in the discovery that no such simulation exists; in fact, T/TCP exhibits user-visible behavior that is not present in TCP, and T/TCP does not even implement the same specification – it can deliver duplicate data to the user at the server end. Discussions with protocol designers suggested that this behavior was not disastrous, so Smith developed a weaker specification that captures the guarantees that T/TCP actually makes. He is currently working on proving that T/TCP satisfies this weaker specification.

Based on his observation of duplicate delivery in T/TCP, Smith considered the question of whether the combination of (correctness and performance) conditions that T/TCP is intended to satisfy can in fact be achieved. He obtained an impossibility result for the *at-most-once fast delivery problem*, an abstract formulation of the problem that T/TCP is designed to solve.
See URL `http://theory.lcs.mit.edu/~mass/comm.html`.
URL `http://theory.lcs.mit.edu/~mass/imposs.html`.

This project on TCP and T/TCP used a combination of timed and untimed automata, with invariants, forward and backward simulations, and embeddings as discussed in Section 3.1.1. It also used the live timed automaton model discussed in Section 2.5, for the impossibility proof; for use in that proof, the model had to be augmented with some structure for describing local clocks.

3.1.3 Other work

Various other communication projects were carried out during the time of the contract, though not specifically in terms of the new formal models. For example, Gawlick worked with Plotkin and Kamath of Stanford and Ramakrishnan of AT&T Bell Labs to develop some admission control and routing algorithms for ATM networks [27, 30, 29]. The algorithms were developed by combining recent theoretical advances with stochastic analysis. Simulations showed the algorithms to perform well in practice.

3.2 Hybrid systems

We carried out an extensive set of case studies in the area of hybrid systems in order to help us to develop and validate our hybrid automaton model. Initially, we spent a good deal of time searching for an appropriate application within the area of hybrid systems – one that was tractable but not trivial, of practical importance, and able to benefit from the use of careful modelling and analysis methods. The application we eventually settled on was that of automated transportation systems.

Our research in automated transportation systems was (and is) intended to build up a collection of techniques for representing, designing, and reasoning about automated control systems. Such techniques should also be useful for military applications such as autonomous weapons systems and semi-automated flight systems.

An overview of the group's work on modelling automated transportation systems, covering (only) the work through October, 1995, appears in [59].
See URL `http://theory.lcs.mit.edu/tds/prt.html`.

3.2.1 Generalized railroad crossing

The first case study was a simple one based on the toy Generalized Railroad Crossing problem proposed by Heitmeyer and others at the Naval Research Laboratory as a challenge problem for evaluating formal methods for modelling and verifying real-time process control systems. Because we felt the stated requirements needed discussion, we ended up working collaboratively with Heitmeyer in developing our solution [34, 35]. We used our timed automaton model and compositional, invariant assertion and simulation relation methods; in particular, in order to prove timing properties, we used the methods described in Section 2.2. We were able to verify all required properties, in complete generality, using parameters for various time bound assumptions. (Most of the other approaches fixed particular values for the assumed time bounds.) We later revised this paper for inclusion as a chapter in a book on formal methods for real-time computing [36].
See URL `http://theory.lcs.mit.edu/tds/grc.html`.
Later, Archer and Heitmeyer at NRL checked most of our proof details mechanically using PVS.

3.2.2 Steam boiler

Our next hybrid systems case study (not about transportation) was another challenge problem, this one for a case study in formal methods for industrial applications. Namely, Leeb and Lynch prepared a paper modelling a steam boiler system [45], and presented this work at a meeting devoted to this problem in June, 1995. See URL `http://theory.lcs.mit.edu/tds/boiler.html`.

Again, Archer and Heitmeyer checked our proof using PVS (this work uncovered some small errors and one significant error, all of which we subsequently fixed) [5].

For this example, we again used timed automata, composition, invariants and simulations. However, we found that although our methods were adequate for the problem at hand, they were not optimal – they did not provide the most suitable facilities for modelling the continuous behavior of the steam boiler (changes in temperature, pressure and volume). This motivated us to develop the more general *hybrid I/O automaton model*, discussed in Section 2.6 above, which provides facilities for directly modelling continuous real-world behavior.

3.2.3 Deceleration maneuver

Weinberg and Lynch applied the timed automaton and hybrid automaton models, and composition, invariant assertion and simulation proof methods, to describe and analyze a collection of typical vehicle deceleration maneuvers [79, 117, 116]. The maneuvers involved reliably reducing the speed of a vehicle to an acceptable limit before reaching a designated track location. This problem was considered with and without feedback, and in the presence of various types of timing uncertainty.

For this work, we initially used timed automata, noted the limitations on expressive power that are discussed in Section 3.2.2 above, and then switched to using hybrid automata. Our proofs treated continuous and discrete facts separately, using different methods (as discussed in Section 2.6). The techniques provided easy proofs for all properties (except that our use of continuous mathematical methods in this work was a bit too brute-force – more on this in Section 3.2.5).
See URL `http://theory.lcs.mit.edu/~hbw/decel.html`.

3.2.4 Vehicle protection systems

Weinberg, Lynch and Delisle (of Raytheon) [118] produced a preliminary model for structure and behavior of the *vehicle protection system* portion of the Raytheon Personal Rapid Transit project. This subsystem interacts with the vehicles and the *vehicle control system* in order to ensure basic safety constrains (e.g., collision avoidance, overspeed protection). They proved that certain of the vehicle protectors in fact guarantee their specified safety properties, even when used in combination (and relying on each other's correct behavior). The correctness proofs use the notion of an "abstract protector" – a generic component that captures the abstract functionality of a protector without considering the particular physical plant and protector details. This work uses hybrid automata and composition, invariant and simulation methods.
See URL `http://theory.lcs.mit.edu/~hbw/prot.html`.

Livadas and Lynch continued this work [49] (still in progress). The continuation of this project involved more complicated track topology, more different types of safety subsystems (e.g., safety

on track merges and diverges), and more complicated interactions among different protectors. It also involved considerable generalization of the abstract protectors developed in [118]. Correctness proofs were completed for protectors preventing overspeed and collisions both for a straight track and a general track topology involving multiple Y-shaped merges and diverges. Some technicalities remain to be addressed.
See URL `http://theory.lcs.mit.edu/~clivadas/research.html`.

3.2.5 Platoons of vehicles

Dolginova and Lynch worked on modular safety analysis for the *platoon join* maneuver of the California PATH automated highway project. They modelled systems of vehicles using hybrid automata, and formulated and proved conditions under which vehicles are "safe", that is, guaranteed not to collide at greater than a prespecified speed. The proofs mainly involve proving invariant assertions (in particular, *safety assertions* describing safe vehicle configurations), using a combination of continuous methods and discrete methods. They also demonstrated, using similar methods, that certain conditions are unsafe.

Upon noting our "brute force" use of continuous methods, Branicky (a control theorist) proposed some more powerful derivative-based techniques, which we used to simplify some of our proofs – see Section 2.6. This work appears in [16, 21].
See URL `http://theory.lcs.mit.edu/tds/platoons.html`.

Dolginova won (shared) the Fano prize for the top MIT undergraduate project in EECS in 1996-1997.

The work described so far in this section only considered the first collision between a pair of vehicles; however, there are safety issues for subsequent collisions as well. Lygeros and Lynch have begun modelling and analyzing multiple collisions in platoons of vehicles. In particular, they are examining the special case of emergency braking of a platoon of vehicles, in the realistic case where the vehicles in the platoon might have different braking capabilities and different masses. They are seeking conditions under which such a maneuver can be executed safely, i.e., so that all collisions occur at acceptably low relative velocities.
See URL `is http://theory.lcs.mit.edu/tds/epm.html`.

This work is intended not just to contribute results about the safety of platoon systems, but also to help establish links between computer science techniques (e.g., invariant assertions and simulation relations) and control theory techniques (e.g., optimal control and continuous game theory) for designing and analyzing hybrid systems.

3.2.6 Multilevel analysis of hybrid systems

Lynch has analyzed a vehicle acceleration maneuver, using three levels of abstraction [60]. The levels capture the relationship between local control and global effects, and also between discrete sampling and continuous feedback. This serves to illustrate two important uses of simulation relation techniques in the context of hybrid systems.
See URL `http://theory.lcs.mit.edu/tds/three-level.html`.

3.2.7 Aircraft control

Many of the methods that we have been developing for automated ground transportation systems appear to apply also for other types of control systems such as aircraft control systems. We have begun a preliminary investigation of this applicability.

Namely, Lygeros has begun considering the problem of verifying that the newly-proposed TCAS conflict detection/resolution algorithm guarantees safety, i.e., that under reasonable assumptions, it maintains a minimum separation between the aircraft [52]. Lygeros has developed a preliminary model for the physical system, and is currently working on modelling the protocol. This work should be important to the area of air-traffic management because it can provide ways of formally proving the correctness of the protocols before they are deployed. This work is also a test of utility for our model and methods and a spur to their further development.
See URL `http://theory.lcs.mit.edu/tds/TCAS.html`.

3.3 Distributed system building blocks

A considerable amount of our group's effort was (and still is) devoted to the identification and analysis of *building blocks* for the construction of efficient, fault-tolerant distributed systems. Ideally, such building blocks should include information about performance and fault-tolerance as well as ordinary correctness properties. In some of the examples listed below, timing aspects were not considered, which means that our untimed models were sufficient. In most, however, timing aspects played an important role, and we used the timed automaton model (at least, for those aspects of the examples that dealt with timing).

3.3.1 Distributed shared memory

In our first effort in this area (for which we did not consider any timing aspects), Fekete, Lynch and Kaashoek [24, 25, 23] modelled the key algorithms used in the Orca system of Bal, Kaashoek and Tanenbaum [13]. The Orca system implements a shared memory service on top of an atomic

broadcast communication service. In carrying out this work, we found a significant logical error in the Orca system, which required some reprogramming. For the corrected system, we produced a nicely decomposed representation (in terms of an intermediate multicast service) and a complete proof.
See URL `http://theory.lcs.mit.edu/tds/orca.html`.

3.3.2 Eventually serializable data service

Fekete, Gupta, Luchangco, Lynch and Shvartsman developed a notion of *eventually serializable data service* [22]; this service relaxes consistency guarantees provided by traditional distributed data services in order to improve system efficiency and availability. The service can be used as a distributed system building block for data service applications that need quick system response and that can tolerate transient inconsistencies in the replies. They have demonstrated the usefulness of the service for describing practical network name services. They have developed a distributed algorithm for implementing this service, based on ideas of Ladin, Liskov, Shrira and Ghemawat [43], and have verified and analyzed the performance of this algorithm.
See URL `http://theory.lcs.mit.edu/~victor_l/eventually-serializable.html`
and `http://theory.lcs.mit.edu/~victor_l/papers/PODC96.html`.
At present, this work appears only in a conference paper and in a manuscript; it remains for us to produce a journal version.

Shvartsman and Cheiner implemented the distributed algorithm of [22], using a LAN of Unix workstations and the MPI message-passing system. Empirical study of this implementation is in progress.
See URL `http://theory.lcs.mit.edu/tds/proto.html`.

3.3.3 Broadcast-convergecast service

Lynch and Shvartsman formulated a specification of a general purpose *broadcast-convergecast communication service*, which delivers a submitted message to a collection of users, awaits responses from a "quorum" of the users, and combines those responses in a convergecast to produce a response for the original sender [69]. The service performs the convergecast internally, using a user-supplied *condenser function* for combining the responses. The service allows the user to specify the quorum configuration, and so permits the use of dynamic quorums. Lynch and Shvartsman used the service to construct two distributed implementations of atomic shared memory, using replicated data-management protocols. One of these is based on dynamic quorums. The algorithms are proved correct (using invariants) and their performance analyzed.
See URL `http://theory.lcs.mit.edu/FTCS97-sub-paper.html`.

3.3.4 Group Communication

Fekete, Lynch, and Shvartsman produced a new and simple formal specification for a *view-synchronous group communication* (VSGC) service similar to group communication services used in systems like Isis, Horus, Transis, and Totem [26]. Our paper contains an untimed automaton specification for the safety aspects of the service, plus a timed trace property specification for the performance and fault-tolerance aspects. This second part is based on the timed automaton model.

Fekete et al. developed an algorithm using VSGC to implement a totally ordered broadcast service, based on a previous algorithm of Dolev and his students [4, 41] that reconciles information derived from different views of the current group of processors. They verified this algorithm using invariants and simulations, and analyzed its performance and fault-tolerance. The performance and fault-tolerance analysis was done for "stabilized" situations, in which the "failure status" of processors and links does not change and in which the non-failed portions of the system exhibit good performance. All the analysis, including that of performance and fault-tolerance, is done in a modular way. Archer has begun work on verifying the safety proofs using PVS.
See URl `http://theory.lcs.mit.edu/tds/vsgc.html`.

Khazan is working with Fekete, Lynch and Shvartsman to model a load-balancing algorithm that also uses VSGC.
See URL `http://theory.lcs.mit.edu/~roger/Research/research.html#DBS`.

3.3.5 Paxos

De Prisco, Lampson and Lynch produced a complete model, proof and analysis for Lamport's Paxos algorithm for fault-tolerant distributed consensus [19, 90], all using timed automata. The algorithm is decomposed into separate pieces (including separate failure-detector, leader-elector and starter components), all modelled as timed automata. The safety proof uses invariants. For performance and fault-tolerance, we used a stabilized analysis based on Clock Timed Automata, as discussed in Section 2.4.
We believe the Paxos algorithm is the most practical algorithm available for fault-tolerant consensus.
See URL: `http://theory.lcs.mit.edu/tds/paxos.html`.

3.3.6 Other work

We carried out several other "building-blocks" projects involving various memory models. These did not involve timing, however:

Vaziri proved correctness for a controller algorithm for the RAID level 5 system [115]. The proof featured a *recoverability* condition for the operation graphs used in the algorithm. This work

helped to clarify previous work on RAID algorithms, uncovered an error in the RAID level 6 design, and identified another situation where RAID level 6 used more constraints on concurrency than necessary.
See URL `http://theory.lcs.mit.edu/~vaziri/raid.html`.

Luchangco developed a theory of precedence-based memory models, which generalize multiple processor memory models, and abstract away system implementation details. He defined a generalized notion of sequential consistency and a weak consistency requirement called *per-location sequential consistency*, and established sufficient conditions under which the two types of memory are indistinguishable to clients. He also proved that an algorithm used by the Cilk system [15] implements a per-location sequentially consistent memory.
See URL `http://theory.lcs.mit.edu/~victor_l/precedence.html`
or URL `http://theory.lcs.mit.edu/~cilk`.

Frigo and Luchangco have begun to develop a theory of "computation-centric memory models", which characterize memories from the programmer's point of view. A computation is a generalization of an instruction stream. Memory models are expressed in terms of these computations, allowing the programmer to reason about what a program specifies rather than about low-level system details. They have defined sequential consistency in this framework, along with several weak consistency models, and have proved some properties of these models, as well as relationships among them.
See URL `http://theory.lcs.mit.edu/~victor_l/computation.html`.

3.4 Probabilistic systems

Our final set of case studies involved probabilistic distributed systems. As mentioned in Section 2.7, we used our probabilistic model and methods on a variety of case studies; these involved complex randomized distributed algorithms from the distributed computing theory research community. The usual proofs and analyses for these algorithms are quite informal, which is problematic in view of the subtlety of the probabilistic claims. The usual source of difficulty in the arguments is the complicated interplay between nondeterministic and probabilistic choice. Our model and methods handle this and other difficulties cleanly.

3.4.1 Dining philosophers

Lynch, Saias and Segala [62] proved correctness of the well-known randomized Dining Philosophers algorithm of Lehmann and Rabin [46]. In [46], this algorithm had only a proof sketch showing eventual progress with probability one, and in fact, it was not clear to us how to turn this sketch into a correct proof. Our proof gives a more refined analysis – a probabilistic time bound – and is done completely formally in terms of our probabilistic model. This proof uses our "coin lemma" technique for reducing the probabilistic system of interest to a non-probabilistic system. See URL `http://theory.lcs.mit.edu/~segala/PODC94.html`.

Segala and Pogosyants applied our probabilistic model and its proof rules to give a computer-assisted correctness and time performance proof for Lehmann and Rabin's algorithm, using the Larch Prover [87]. This proof also used coin lemmas to reduce the probabilistic system to a non-probabilistic one, and then used known automatic techniques on the resulting non-probabilistic system. See URL `http://theory.lcs.mit.edu/~segala/PS95.html`.

3.4.2 Network spanning tree

Aggarwal, Lynch and Segala [1] proved correctness of a new and subtle self-stabilizing network spanning tree algorithm of Aggarwal and Kutten [3], exposing and fixing a bug in the process. The proof is based on progress statements and coin lemmas. A feature of this proof is that it manages to isolate the probabilistic reasoning to only a very small portion of the paper – most of the argument involves standard non-probabilistic analysis.
See URL `http://theory.lcs.mit.edu/TR-632.html`.

3.4.3 Randomized consensus

Pogosyants and Segala modelled and analyzed Ben-Or's randomized consensus protocol [94], using coin lemmas and generalized versions of progress statements that deal with generalized complexity measures (rather than just with time). The use of general complexity measures turned out to be convenient for the complexity analysis of asynchronous algorithms.
See URL `http://theory.lcs.mit.edu/~segala/phd.html`.

Pogosyants, Segala and Lynch used random walk methods and modular techniques for time performance analysis, as described in Section 2.7, to model and analyze the very complex randomized consensus protocol of Aspnes and Herlihy [88, 89]. Again, the probabilistic part of the reasoning is confined to a few short sections of the paper. Most of the reasoning involves invariants. The proof is highly modular, and comparable in length to the original (less formal) analysis of Aspnes and Herlihy. The development of the proof led to the following new verification techniques: new coin lemmas for random walks, rules for proving probabilistic properties of a complex system based on probabilistic properties of one of its components, rules for combining different complexity measures, and rules for deriving relations between expected complexity bounds based on relations between complexity measures. This last kind of rule is very important since, once again, it allows us to reduce a probabilistic problem to a nondeterministic problem.

This example demonstrates that our methods are usable for the analysis of even the most complex existing randomized algorithms.
See URL `http://theory.lcs.mit.edu/tds/AH.html`.

4 Algorithms

In addition to its work on modelling and case studies, described above, our group carried out a considerable amount of research on distributed algorithms and impossibility results. We summarize this work in this section. Some of these results involve timing and some do not. Of the results that involves timing, some are described explicitly in terms of timed automata, and some treat the timing model less formally (in the style typical for algorithms papers); however, these could be expressed formally in terms of timed automata.

Our algorithms work falls generally into the categories of communication protocols, data structures supporting efficient concurrent access, fault-tolerant asynchronous computability, clock synchronization, and "other work". Two new books on algorithms were also produced.

4.1 Communication protocols

4.1.1 Connection management protocols

Kleinberg, Lynch, and Attiya proved tradeoff lower bounds for the message delivery time vs. the quiesce time for connection-management protocols in a timing-based setting [42].
See URL `http://theory.lcs.mit.edu/tds/ISTCS95.html`.

As described in Section 3.1.2, Smith proved an impossibility result for the "at-most-once fast delivery problem", an abstract formulation of the problem that T/TCP is designed to solve [110]. This work used the live timed automaton model discussed in Section 2.5, augmented with local clocks.
See `http://theory.lcs.mit.edu/~mass/imposs.html`.

4.1.2 On-line virtual circuit routing

Gawlick wrote a PhD thesis containing a collection of results in the area of on-line virtual circuit routing [29]. In particular, with Kalmanek and Ramakrishnan of AT&T Bell Labs, he developed some new algorithms for routing permanent virtual circuits, which model circuits that are leased by businesses to construct private networks [30]. With Ramakrishnan and with Plotkin and Kamath of Stanford University, Gawlick worked on routing and admission control algorithms for switched

virtual circuits, which model circuits that have a relatively short holding time, such as phone calls, video conference calls, home movies, etc. A key focus of this effort was to design a *distributed* routing protocol [27]. Gawlick also collaborated with Awerbuch and with Azar of Tel Aviv University to develop routing and admission control algorithms for multicast connections [9]. Finally, working with Awerbuch, Leighton and Rabani, Gawlick investigated some theoretical aspects of admission control and routing algorithms for tree, mesh, and hypercube networks [10].
See URL `http://theory.lcs.mit.edu/~rgawlick/phd.html`.

4.2 Concurrent data structures

In this section, we describe work that was led by Prof. Nir Shavit, a visiting professor from Tel Aviv University working in the TDS group. This work involves the design of data structures supporting efficient, highly concurrent access. It has yielded algorithms for interprocess communication and synchronization that have mathematically-provable computability and resiliency properties, and also run efficiently in experiments on actual and simulated multiprocessor machines. These data structures are intended for computing environments ranging from tightly coupled multiprocessors to farms of workstations in local area networks.

Traditionally, the design of concurrent data structures has been based on *mutual exclusion*, which ensures that only one processor at a time is allowed to access a complex data structure. Shavit's data structures allow more concurrency, although fine-grain critical sections are still used by processors at specified *coordination points*. The resulting approach has already yielded structures such as diffracting trees, stacks, and pools, which experimentally outperform conventional solutions.

During the period of the AFOSR contract, Shavit and co-workers continued their work on *diffracting trees* [107, 108]. Diffracting trees are novel data structures used to accomplish shared counting and load balancing; they are based on the *counting network* approach [7, 37] introduced a few years ago. They can be used to construct efficient shared queues, stacks and pools.
See URL `http://theory.lcs.mit.edu/tds/dds.html` and
`http://theory.lcs.mit.edu/~asaph`.

Shavit and Touitou [103, 99] developed a special type of diffracting tree called an *elimination tree*, which utilizes matching operations (such as enqueue/dequeue) to help in constructing efficient shared stacks and pools.
See URL `http://theory.lcs.mit.edu/~shanir/st95.ps`.

Their empirical performance data shows that diffracting trees and elimination pools substantially outperform all previously known techniques: they scale better, giving higher throughput over a large number of processors, and are more robust in terms of their ability to handle unexpected latencies and varying loads.

Shavit, Zemach, and Upfal of IBM developed [100, 101] a stochastic model of diffracting trees that allows certain parameters that govern tree performance to be predicted as a function of the number of processors that are likely to use the structure.
See URL `http://theory.lcs.mit.edu/~shanir/suz.ps`.

This modelling led to a more efficient tree implementation.
See URL `http://theory.lcs.mit.edu/~asaph` and
URL `http://theory.lcs.mit.edu/tds/dds.html`.

Lynch, Shavit, Shvartsman, and Touitou proved that, under reasonable timing constraints, several classes of highly concurrent data structures (such as diffracting trees and counting networks) exhibit linearizable behavior [68]. Touitou and Shvartsman carried out a suite of simulations validating the theoretical results. A journal paper is being prepared.
See URL `http://theory.lcs.mit.edu/~alex/count2.html`.

Della Libera and Shavit developed a *reactive diffracting tree* protocol [18, 47]. This protocol allows the tree to shrink and grow based on the load, closely following the optimal tree size for a given number of processors. This improves performance at low loads so that in effect the tree is like a simple "queue lock" at low loads, with the ability to grow into a powerful diffracting tree as the load increases.
See `http://theory.lcs.mit.edu/~gio/research.html`.

Shavit, Upfal, and Zemach also developed a new "wait-free" sorting algorithm [102], that is, one that takes logarithmic parallel time and still runs (though slightly less effectively) even if many processes fail.
See `http://theory.lcs.mit.edu/~shanir/suz97.ps`.

4.3 Fault-tolerant asynchronous computability

Several of our projects involved attempts to classify problems according to their computability or time complexity in fault-prone asynchronous distributed systems:

Rajsbaum worked with Attiya and Herlihy [8, 38], on characterizing the problems that can be solved in fault-prone asynchronous systems. Herlihy and Rajsbaum [38] analyzed solvability of the fundamental k-consensus problem (wherein processes that start with arbitrary inputs have to agree on a total of at most k final values) in terms of various common types of objects (read/write, test-and-set, fetch-and-add, etc.); this work uses techniques of algebraic topology. Attiya and Rajsbaum [8] developed a characterization theory that is based on elementary combinatorics rather than topology.
See URL `http://theory.lcs.mit.edu/~rajsbaum`.

Lynch and Rajsbaum [61] carried out a careful treatment of an exciting recent idea of Borowsky and Gafni – a fault-tolerant simulation algorithm that allows shared memory algorithms for certain decision problems to be used to solve other decision problems, with the same fault-tolerance properties. Lynch and Rajsbaum's main contribution was to convert the intuitive ideas to a well-defined algorithm, with well-defined correctness properties and a real correctness proof; this was far from a straightforward task. A journal paper is in preparation. This work was all carried out carefully in terms of our untimed I/O automaton model.
See URL `http://theory.lcs.mit.edu/tds/borowsky.html`.

Hoest and Shavit developed a novel mathematical model for evaluating the complexity of algorithms in an asynchronous setting, based on techniques of algebraic topology. They used their methods to analyze time complexity in the *iterated immediate snapshot* model, a restricted type of atomic snapshot shared memory model. They obtained tight bounds for the approximate agreement problem, and a fundamental time vs. number of names tradeoff for the process renaming problem. This work appears in [39]. Hoest and Shavit are currently working on extending their complexity theory to other types of shared memory models.
See URL `http://theory.lcs.mit.edu/~gunnar/acplx.html`.

Chlebus, De Prisco and Shvartsman developed a new fault-tolerant algorithm for the *Do-All* problem of performing n tasks using p message-passing processors under the constraint of maintaining message and work efficiency. This is the first algorithm for the problem that efficiently deals with processor restarts. A technical report documenting this work was submitted for publication [17].
See URL `http://theory.lcs.mit.edu/~alex/cds97.html`.

4.4 Clock synchronization

Patt-Shamir completed his PhD thesis [84] on the topic of clock synchronization; some of this work was carried out jointly with Rajsbaum [85]. This work included the definition of a model for the problem of synchronizing geographically distributed clocks, built upon the timed automaton model. They developed new techniques for analyzing clock synchronization algorithms, and used their methods to obtain algorithms that achieve the optimal degree of synchronization, for any pattern of communication. See URL `http://www.ccs.neu.edu/home/boaz/thesis-abs.html`.

4.5 Other work

Patt-Shamir, Awerbuch and Varghese [11] devised a general method for transforming unbounded register protocols so that they can work with bounded registers, and in a self-stabilizing fashion. They demonstrated the applicability of their method with new algorithms for the problems of spanning tree computation and topology update. In [12] they presented a general paradigm, based

on local chacking and global reset, for making asynchronous network protocols self-stabilizing.
See URL `http://www.ccs.neu.edu/home/boaz/unbounded-abs.html`
or URL `http://www.ccs.neu.edu/home/boaz/ss-compilation-abs.html`.

Other algorithms and lower bound results were also developed in our group during the period of the contract; these are included in Attachment A.

4.6 Books

Lynch wrote an 800-page graduate text book entitled *Distributed Algorithms* [58]. It presents the basic results (algorithms and impossibility results) of the research area of Distributed Algorithms, all unified in terms of our basic untimed and timed automaton models. This unification turned out to be a major research effort. The chapters connected most closely to the AFOSR project are Chapters 23-25, which present the timed automaton model and use it to explain results about mutual exclusion and distributed consensus in networks satisfying certain timing assumptions.
See URL `http://theory.lcs.mit.edu/tds/distalgs.html`.

Shvartsman produced a book entitled *A Theory of Fault-Tolerant Parallel Computation* [40], which contains a synthesis of the latest results about parallel computation in the presence of failures and delays. The monograph deals with several models of processor failures and restarts, it identifies the key problems to be solved in these models, and presents algorithms, general simulations and lower bounds. See URL `http://theory.lcs.mit.edu/~alex/mono2.html`.

5 Tools

5.1 IOA programming language

Garland and Lynch are completing a preliminary design of a programming language, IOA, for our untimed I/O automaton model. IOA allows simple abstract description of distributed systems, and is intended to aid in distributed system development, testing and verification, all in one coordinated framework. We intend for the IOA language to be integrated with a variety of tools, including simulators, theorem-provers, and model-checkers. Eventually, generation of distributed code should be possible.

Garland, Lynch and Vaziri are writing a user's manual for IOA. The language is formulated in terms of Larch, and will allow use of the Larch Prover for verification; however, we intend that the language will also be usable with other theorem provers such as PVS.

We also plan to connect the language to existing model-checking tools; as a start in this direction, Petrov and Vaziri worked on a translation scheme from IOA to the input language of the model

checker SPIN.
See URL `http://theory.lcs.mit.edu/~petrov/IOAtoPROMELA.html`.

We also plan a simulator, and eventually, hope to support real distributed code-generation as well (via programming in levels of abstraction and translation of the lowest level of IOA to an existing language such as C++ or Java).
See URL `http://larch.lcs.mit.edu:8001/~garland/ioaLanguage.html`.

Note that IOA does not include timing features – this first cut is based just on our untimed I/O automaton model. However, if this first attempt works well, the obvious next step is to introduce timing into the language and tools.

References

[1] Sudhanshu Aggarwal. Time optimal self-stabilizing spanning tree algorithms. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, May 1994. Also, [2].

[2] Sudhanshu Aggarwal. Time optimal self-stabilizing spanning tree algorithms. Technical Report MIT/LCS/TR-632, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, May 1994. Also, [1].

[3] Sudhanshu Aggarwal and Shay Kutten. Time optimal self stabilizing spanning tree algorithms. In R.K. Shyamasundar, editor, *13th International Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 761 of *Lecture Notes in Computer Science*, pages 400–410, Bombay, India., December 1993. Springer-Verlag.

[4] Y. Amir, D. Dolev, P. Melliar-Smith, and L. Moser. Robust and efficient replication using group communication. Technical Report 94-20, Department of Computer Science, Hebrew University, 1994.

[5] Myla Archer and Constance Heitmeyer. Verifying hybrid systems modeled as timed automata: A case study. In Oded Maler, editor, *Hybrid and Real-Time Systems* (International Workshop, HART'97, Grenoble, France, March 1997), volume 1201 of *Lecture Notes in Computer Science*, pages 171–185, Berlin Heidelberg, 1997. Springer-Verlag.

[6] James Aspnes and Maurice Herlihy. Fast randomized consensus using shared memory. *Journal of Algorithms*, 11(3):441–461, September 1990.

[7] James Aspnes, Maurice Herlihy, and Nir Shavit. Counting networks. *Journal of the ACM*, 41(5):1020–1048, September 1994. Also, Earlier version in *Proceedings of the 23rd ACM*

Annual Symposium on Theory of Computing, pp. 348–358, May 1991. Also, MIT Technical Report MIT/LCS/TM-451, June 1991.

[8] Hagit Attiya and Sergio Rajsbaum. A combinatorial framework for wait-free computability. Technical Report 95/3, Digital Equipment Corporation, Cambridge Research Lab, Cambridge, MA 02139, March 1995.

[9] B. Awerbuch, Y. Azar, and R. Gawlick. Competitive on-line routing and admission control for multicast. Manuscript, 1994.

[10] Baruch Awerbuch, Rainer Gawlick, Tom Leighton, and Yuval Rabani. On-line admission control and circuit routing for high performance computation and communication. In *35th Annual Symposium on Foundations of Computer Science*, pages 412–423, Santa Fe, New Mexico, November 1994. IEEE.

[11] Baruch Awerbuch, Boaz Patt-Shamir, and George Varghese. Bounding the unbounded. In *Proceedings of the 13th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '94)*, pages 776–783, Toronto, Ontario, June 1994.

[12] Baruch Awerbuch, Boaz Patt-Shamir, and George Varghese. Self-stabilization by local checking and global reset. In *Proceedings of the 8th International Workshop on Distributed Algorithms (WDAG'94)*, volume 857 of *Lecture Notes in Computer Science*, pages 326–339, Terschelling, The Netherlands, September 1994.

[13] H. E. Bal, M. F. Kaashoek, and A. S. Tanenbaum. Orca: A language for parallel programming of distributed systems. *IEEE Trans. on Soft Eng.*, 18(3):190–205, March 1992.

[14] Dag Belsnes. Single-message communication. *IEEE Transactions on Communications*, COM-24(2):190–194, February 1976.

[15] Robert D. Blumofe, Christopher F. Joerg, Bradley C. Kuszmaul, Charles E. Leiserson, Keith H. Randall, and Yuli Zhou. Cilk: An efficient multithreaded runtime system. In *Proceedings of the Fifth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, pages 207–216, Santa Barbara, California, July 1995.

[16] Michael S. Branicky, Ekaterina Dolginova, and Nancy Lynch. A toolbox for proving and maintaining hybrid specifications. Presented at *HS'96: Hybrid Systems*, October 12-16, 1996, Cornell University, Ithacs, NY.

[17] Bogdan Chlebus, Roberto DePrisco, and Alex Shvartsman. Performing tasks on restartable message-passing processors. Submitted for publication.

[18] Giovanni Della-Libera. Dynamic diffracting trees. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, July 1996.

[19] Roberto DePrisco, Butler Lampson, and Nancy Lynch. Revisiting the Paxos algorithm. Manuscript, January 1997.

[20] Danny Dolev and Nir Shavit. Bounded time stamping. *SIAM Journal on Computing*, 26(2):418–455, April 1997.

[21] Ekaterina Dolginova and Nancy Lynch. Safety verification for automated platoon maneuvers: A case study. In Oded Maler, editor, *Hybrid and Real-Time Systems* (International Workshop, HART'97, Grenoble, France, March 1997), volume 1201 of *Lecture Notes in Computer Science*, pages 154–170. Springer-Verlag, 1997.

[22] Alan Fekete, David Gupta, Victor Luchangco, Nancy Lynch, and Alex Shvartsman. Eventually-serializable data services. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 300–309, Philadelphia, PA, May 1996.

[23] Alan Fekete, M. Frans Kaashoek, and Nancy Lynch. Implementing sequentially consistent shared objects using broadcast and point-to-point communication. Submitted for journal publication.

[24] Alan Fekete, M. Frans Kaashoek, and Nancy Lynch. Implementing sequentially consistent shared objects using broadcast and point-to-point communication. In *Proceedings of the 15th International Conference on Distributed Computing Systems (ICDCS'95)*, pages 439–449, Vancouver, Canada, May/June 1995. IEEE.

[25] Alan Fekete, M. Frans Kaashoek, and Nancy Lynch. Implementing sequentially consistent shared objects using broadcast and point-to-point communication. Technical Memo MIT/LCS/TM-518, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, June 1995.

[26] Alan Fekete, Nancy Lynch, and Alex Shvartsman. Specifying and using a partitionable group communication service. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA, August 1997. To appear.

[27] R. Gawlick, A. Kamath, S. Plotkin, and K. Ramakrishnan. Routing and admission control of virtual circuits in general topology networks. Submitted for publication, 1995.

[28] Rainer Gawlick. Bounded concurrent time-stamping made simple. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 1992.

[29] Rainer Gawlick. *Admission Control and Routing: Theory and Practice.* PhD thesis, Department of Electrical Engineering and Computer Science, June 1995. Also, MIT/LCS/TR-679.

[30] Rainer Gawlick, Charles Kalmanek, and K. G. Ramakrishnan. On-line routing for permanent virtual circuits. *Computer Communications*, 19:235–244, 1996. Previous version appeared in *Proceedings of IEEE INFOCOM 95: Fourteenth Annual Joint Conference of the IEEE Computer and Communication Societies*, pages 278-288, Boston, Massachusetts, April 1995.

[31] Rainer Gawlick, Nancy Lynch, and Nir Shavit. Concurrent time-stamping made simple, 1995. Full version submitted for journal publication. Earlier version appears in *Proceedings of the First Israel Symposium on the Theory of Computing and Systems*, Springer-Verlag, pages 171-185, May 1992.

[32] Rainer Gawlick, Roberto Segala, Jørgen Søgaard-Andersen, and Nancy Lynch. Liveness in timed and untimed systems. Technical Report MIT/LCS/TR-587, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139, December 1993.

[33] Rainer Gawlick, Roberto Segala, Jørgen Søgaard-Andersen, and Nancy Lynch. Liveness in timed and untimed systems. In Serge Abiteboul and Eli Shamir, editors, *Automata, Languages and Programming* (21st International Colloquium, ICALP'94, Jerusalem, Israel, July 1994), volume 820 of *Lecture Notes in Computer Science*, pages 166–177. Springer-Verlag, 1994. Full version in [32]. Also, submitted for publication.

[34] Constance Heitmeyer and Nancy Lynch. The generalized railroad crossing: A case study in formal verification of real-time systems. In *Proceedings of the Real-Time Systems Symposium*, pages 120–131, San Juan, Puerto Rico, December 1994. IEEE.

[35] Constance Heitmeyer and Nancy Lynch. The generalized railroad crossing: A case study in formal verification of real-time systems. Technical Memo MIT/LCS/TM-511, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, November 1994.

[36] Constance Heitmeyer and Nancy Lynch. Formal verification of real-time systems using timed automata. In Constance Heitmeyer and Dino Mandrioli, editors, *Formal Methods for Real-Time Computing*, Trends in Software, chapter 4, pages 83–106. John Wiley & Sons Ltd, April 1996.

[37] M. Herlihy, B. H. Lim, and N. Shavit. Scalable concurrent counting. *ACM Transactions on Computer Systems*, 13(4):343–364, 1995. Earlier version in *Proceedings of the Third Annual ASM Symposium on Parallel Algorithms and Architectures (SPAA)*, San Diego, CA, pages 219-227, July 1992. Full version available as DEC TR.

[38] Maurice Herlihy and Sergio Rajsbaum. Algebraic spans. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 90–99, Ottawa, Ontario, Canada, August 1995.

[39] Gunnar Hoest and Nir Shavit. Towards a topological characterization of asynchronous complexity. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA, August 1997. To appear.

[40] Paris C. Kanellakis and Alex A. Shvartsman. *A Theory of Fault-Tolerant Parallel Computation.* Kluwer Academic Publishers, Boston, MA, 1997.

[41] I. Keidar and D. Dolev. Efficient message ordering in dynamic networks. In *Proceedings of the 15th Annual ACM Symposium on Distributed Computing*, pages 68–76, Philadelphia, PA, May 1996.

[42] Jon Kleinberg, Hagit Attiya, and Nancy Lynch. Trade-offs between message delivery and quiesce times in connection management protocols. In *Proceedings of ISTCS 1995: The Third Israel Symposium on Theory of Computing and Systems*, pages 258–267, Tel-Aviv, Israel, January 1995. IEEE.

[43] R. Ladin, B. Liskov, L. Shrira, and S. Ghemawat. Exploiting the semantics of distributed services. *ACM Transactions on Computer Systems*, 10(4):360–391, November 1992.

[44] Butler W. Lampson, Nancy A. Lynch, and Jørgen F. Søgaard-Andersen. Correctness of at-most-once message delivery protocols. In Richard L. Tenney, Paul D. Amer, and M. Ümit Uyar, editors, *Formal Description Techniques, VI* (Proceedings of the IFIP TC6/WG6.1 Sixth International Conference on Formal Description Techniques — FORTE'93, Boston, MA, October 1993), pages 385–400. North-Holland, 1994.

[45] Gunter Leeb and Nancy Lynch. Proving safety properties of the steam boiler controller: Formal methods for industrial applications: A case study, 1996. To appear in *Lecture Notes in Computer Science*, Springer-Verlag series. Earlier version presented at the *Methods for Semantics and Specification* (International Conference and Research Center for Computer Science, Schloss, Dagstuhl, Germany, June 1995), as "Using Timed Automata for the Steam Boiler Controller Problem.".

[46] Daniel Lehmann and Michael O. Rabin. On the advantages of free choice: a symmetric and fully distributed solution to the Dining Philosophers problem. In *Proceedings of Eighth Annual ACM Symposium on Principles of Programming Languages*, pages 133–138, Williamsburg, Virginia, January 1981.

[47] Giovanni Della Libera and Nir Shavit. Reactive diffracting trees. In *Proceedings of the 9th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, Newport, Rhode Island, June 1997. To appear.

[48] B. Liskov, Luiba Shrira, and John Wroclawski. Efficient at-most-once messages based on synchronized clocks. Technical Report MIT/LCS/TR-476, Laboratory for Computer Science, Massachusetts Institute of Technology, April 1990.

[49] Carolos Livadas. Verification of automated vehicle protection systems. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, 1996. In progress.

[50] Victor Luchangco. Using simulation techniques to prove timing properties. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, June 1995.

[51] Victor Luchangco, Ekrem Söylemez, Stephen Garland, and Nancy Lynch. Verifying timing properties of concurrent algorithms. In Dieter Hogrefe and Stefan Leue, editors, *Formal Description Techniques VII: Proceedings of the 7th IFIP WG6.1 International Conference on Formal Description Techniques* (FORTE'94, Berne, Switzerland, October 1994), pages 259–273. Chapman and Hall, 1995.

[52] John Lygeros and Nancy Lynch. Formal verification of the TCAS conflict resolution algorithms. In *1997 Conference on Decision and Control*, San Diego, CA, December 1997. To appear. Extended abstract.

[53] Nancy Lynch. Simulation techniques for proving properties of real-time systems. Technical Memo MIT/LCS/TM-494, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, November 1993.

[54] Nancy Lynch. Simulation techniques for proving properties of real-time systems. In J. W. de Bakker, W. P. de Roever, and G. Rozenberg, editors, *A Decade of Concurrency: Reflections and Perspectives* (REX School/Symposium, Noordwijkerhout, The Netherlands, June 1993), volume 803 of *Lecture Notes in Computer Science*, pages 375–424. Springer-Verlag, 1994.

[55] Nancy Lynch. Modelling and verification of automated transit systems, using timed automata, invariants and simulations. Technical Memo MIT/LCS/TM-545, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, December 1995. Also, [59].

[56] Nancy Lynch. Proving performance properties (even probabilistic ones). In Dieter Hogrefe and Stefan Leue, editors, *Formal Description Techniques VII: Proceedings of the 7th IFIP*

WG6.1 International Conference on Formal Description Techniques (FORTE'94), pages 3–20. Chapman and Hall, 1995. Invited talk.

[57] Nancy Lynch. Simulation techniques for proving properties of real-time systems. In Sang H. Son, editor, *Advances in Real-Time Systems*, chapter 13, pages 299–332. Prentice Hall, Inc., Englewood Cliffs, NJ, 1995.

[58] Nancy Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, March 1996.

[59] Nancy Lynch. Modelling and verification of automated transit systems, using timed automata, invariants and simulations. In R. Alur, T. Henzinger, and E. Sontag, editors, *Hybrid Systems III: Verification and Control* (DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, New Brunswick, New Jersey, October 1995), volume 1066 of *Lecture Notes in Computer Science*, pages 449–463. Springer-Verlag, 1996. Also, [55].

[60] Nancy Lynch. A three-level analysis of a simple acceleration maneuver, with uncertainties. In *Proceedings of the Third AMAST Workshop on Real-Time Systems*, pages 1–22, Salt Lake City, Utah, March 1996.

[61] Nancy Lynch and Sergio Rajsbaum. On the Borowsky-Gafni simulation algorithm. In *Proceedings of the Fourth ISTCS: Israel Symposium on Theory of Computing and Systems*, pages 4–15, Jerusalem, Israel, June 1996. IEEE Computer Society. Also, short version appears in *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, Philadelphia, PA, page 57, May 1996.

[62] Nancy Lynch, Isaac Saias, and Roberto Segala. Proving time bounds for randomized distributed algorithms. In *Proceedings of the Thirteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 314–323, Los Angeles, California, August 1994.

[63] Nancy Lynch and Roberto Segala. A comparison between simulation techniques and algebraic techniques for verifying concurrent systems. In *NAPAW Proceedings of the North American Process Algebra Workshop*, Department of Computer Science, Cornell University, Ithaca, NY, August 1993. TR 93-1369. Also, [64], [65].

[64] Nancy Lynch and Roberto Segala. A comparison of simulation techniques and algebraic techniques for verifying concurrent systems. Technical Memo MIT/LCS/TM-499, Laboratory for Computer Science, Massachusetts Institute of Technology, December 1993.

[65] Nancy Lynch and Roberto Segala. A comparison of simulation techniques and algebraic techniques for verifying concurrent systems. *Formal Aspects of Computing*, 7(3):231–265, 1995. Also, [64] and [63].

[66] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H. B. Weinberg. Hybrid I/O automata. Technical Memo MIT/LCS/TM-544, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, December 1995. Also, [67].

[67] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H. B. Weinberg. Hybrid I/O automata. In R. Alur, T. Henzinger, and E. Sontag, editors, *Hybrid Systems III: Verification and Control* (DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, New Brunswick, New Jersey, October 1995), volume 1066 of *Lecture Notes in Computer Science*, pages 496–510. Springer-Verlag, 1996. Also, [66].

[68] Nancy Lynch, Nir Shavit, Alex Shvartsman, and Dan Touitou. Counting networks are practically linearizable. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 280–289, Philadelphia, PA, May 1996.

[69] Nancy Lynch and Alex Shvartsman. Robust emulation of shared memory using dynamic quorum-acknowledged broadcasts. In *Twenty-Seventh Annual International Symposium on Fault-Tolerant Computing (FTCS'97)*, Seattle, Washington, USA, June 1997. To appear.

[70] Nancy Lynch and Frits Vaandrager. Forward and backward simulations for timing-based systems. In J. W. de Bakker et al., editors, *Real-Time: Theory in Practice* (REX Workshop, Mook, The Netherlands, June 1991), volume 600 of *Lecture Notes in Computer Science*, pages 397–446. Springer-Verlag, 1992.

[71] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part I: Untimed systems. Technical Memo MIT/LCS/TM-486, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139, May 1993.

[72] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part II: Timing-based systems. Technical Memo MIT/LCS/TM-487.b, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139, April 1993.

[73] Nancy Lynch and Frits Vaandrager. Action transducers and timed automata. Technical Memo MIT/LCS/TM-480.b, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, October 1994.

[74] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part I: Untimed systems. Technical Memo MIT/LCS/TM-486.b, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139, August 1994.

[75] Nancy Lynch and Frits Vaandrager. Action transducers and timed automata. Technical Memo MIT/LCS/TM-480.c, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, July 1995.

[76] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part I: Untimed systems. *Information and Computation*, 121(2):214–233, September 1995. Also, [71].

[77] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part II: Timing-based systems. Technical Memo MIT/LCS/TM-487.c, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, April 1995.

[78] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part II: Timing-based systems. *Information and Computation*, 128(1):1–25, July 1996. Also, [77].

[79] Nancy Lynch and H. B. Weinberg. Proving correctness of a vehicle maneuver: Deceleration. In *Second European Workshop on Real-Time and Hybrid Systems*, pages 196–203, Grenoble, France, May/June 1995. Later version appears as [117].

[80] Nancy A. Lynch and Hagit Attiya. Using mappings to prove timing properties. *Distributed Computing*, 6(2):121–139, September 1992.

[81] Nancy A. Lynch and Mark R. Tuttle. An introduction to input/output automata. *CWI-Quarterly*, 2(3):219–246, September 1989. Centrum voor Wiskunde en Informatica, Amsterdam, The Netherlands. Technical Memo MIT/LCS/TM-373, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, November 1988.

[82] Nancy A. Lynch and Frits W. Vaandrager. Action transducers and timed automata. *Formal Aspects of Computing*, 8(5):499–538, 1996. Also, [75].

[83] R. De Nicola and R. Segala. A process algebraic view of I/O automata. *Theoretical Computer Science*, 138:391–423, March 1995. Also, Rapporto Tecnico N. SI-92/05, Universitá "La Sapienza", Rome.

[84] Boaz Patt-Shamir. *A Theory of Clock Synchronization.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, October 1994. Also, MIT/LCS/TR-680.

[85] Boaz Patt-Shamir and Sergio Rajsbaum. A theory of clock synchronization. In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, pages 810–819, Montreal, Canada, May 1994. Journal version in progress.

[86] Tsvetomir P. Petrov, Anna Pogosyants, Stephen J. Garland, Victor Luchangco, and Nancy A. Lynch. Computer-assisted verification of an algorithm for concurrent timestamps. In Reinhard Gotzhein and Jan Bredereke, editors, *Formal Description Techniques IX: Theory, Applications, and Tools* (FORTE/PSTV'96: Joint International Conference on Formal Description

Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification, Kaiserslautern, Germany, October 1996), pages 29–44. Chapman & Hall, 1996.

[87] Anna Pogosyants and Roberto Segala. Formal verification of timed properties of randomized distributed algorithms. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 174–183, Ottawa, Ontario, Canada, August 1995.

[88] Anna Pogosyants, Roberto Segala, and Nancy Lynch. Verification of the randomized consensus algorithm of Aspnes and Herlihy: a case study. Manuscript. Also, fuller version submitted for journal publication. Also, [89].

[89] Anna Pogosyants, Roberto Segala, and Nancy Lynch. Verification of the randomized consensus algorithm of Aspnes and Herlihy: a case study. Technical Memo MIT/LCS/TM-555, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, June 1997. To appear.

[90] Roberto De Prisco. Revisiting the Paxos algorithm. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, June 1997. Also, to be MIT/LCS/TM-717, and [19].

[91] Alain Isaac Saias. *Randomness versus Non-Determinism in Distributed Computing.* PhD thesis, MIT Mathematics Department, October 1994. Also, MIT/LCS/TR-651.

[92] Roberto Segala. Quiescence, fairness, testing and the notion of implementation. In Eike Best, editor, *Proceedings of CONCUR 93: 4th International Conference on Concurrency Theory*, volume 715 of *Lecture Notes in Computer Science*, pages 324–338, Hildesheim, Germany, August 1993. Springer-Verlag.

[93] Roberto Segala. A compositional trace-based semantics for probabilistic automata. In Insup Lee and Scott A. Smolka, editors, *CONCUR 95: Concurrency Theory* (6th International Conference, Philadelphia, Pennsylvania, August 1995), volume 962 of *Lecture Notes in Computer Science*, pages 234–248. Springer-Verlag, 1995.

[94] Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, May 1995. Also, MIT/LCS/TR-676.

[95] Roberto Segala. Quiescence, fairness, testing and the notion of implementation. *Information and Computation*, 1997. To appear. Earlier version in [92].

[96] Roberto Segala, Rainer Gawlick, Jørgen Søgaard-Andersen, and Nancy Lynch. Liveness in timed and untimed systems. Submitted for journal publication.

[97] Roberto Segala and Nancy Lynch. Probabilistic simulations for probabilistic processes. In Bengt Jonsson and Joachim Parrow, editors, *CONCUR'94: Concurrency Theory* (5th International Conference, Uppsala, Sweden, August 1994), volume 836 of *Lecture Notes in Computer Science*, pages 481–496. Springer-Verlag, 1994. A revised version appears in the *Nordic Journal of Computing,* special issue on selected papers from CONCUR94, volume 2, number 2, pages 250-273, 1995.

[98] Roberto Segala and Nancy Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, August 1995. Special issue on selected papers from *CONCUR94.*

[99] N. Shavit and D. Touitou. Software transactional memory. *Distributed Computing*, 10(2):99–116, February 1997. Also, [105], and earlier version in [104].

[100] N. Shavit, E. Upfal, and A. Zemach. A steady state analysis of diffracting trees. In *Proceedings of the 8th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 33–41, Padua, Italy, June 1996.

[101] N. Shavit, E. Upfal, and A. Zemach. A steady state analysis of diffracting trees. *Theory of Computing Systems (formerly Mathematical Systems Theory)*, 1997. Special issue. To appear.

[102] N. Shavit, E. Upfal, and A. Zemach. A wait-free sorting algorithm. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA, August 1997. To appear.

[103] Nir Shavit and Dan Touitou. Elimination trees and the construction of pools and stacks. In *SPAA'95: 7th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 54–63, Santa Barbara, California, July 1995. Also, [106].

[104] Nir Shavit and Dan Touitou. Software transactional memory. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 204–213, Ottawa, Ontario, Canada, August 1995. Also, [99], and [105].

[105] Nir Shavit and Dan Touitou. Software transactional memory. Technical Memo MIT/LCS/TM-675, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, 1996. Also, [99] and [104].

[106] Nir Shavit and Dan Touitou. Elimination trees and the construction of pools and stacks. *Theory of Computing Systems (formerly Mathematical Systems Theory)*, 1997. To appear. Earlier version as [103].

[107] Nir Shavit and Asaph Zemach. Diffracting trees. In *Proceedings of the Sixth Annual Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 167–176, Cape May, New Jersey, June 1994.

[108] Nir Shavit and Asaph Zemach. Diffracting trees. *ACM Transactions on Computer Systems*, 14(4):385–428, November 1996. Also, [107].

[109] Mark Smith. Formal verification of communication protocols. In Reinhard Gotzhein and Jan Bredereke, editors, *Formal Description Techniques IX: Theory, Applications, and Tools* FORTE/PSTV'96: Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification, Kaiserslautern, Germany, October 1996, pages 129–144. Chapman & Hall, 1996.

[110] Mark Smith. *Formal Verification of TCP and T/TCP.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, 1997. In progress.

[111] Jørgen Søgaard-Andersen. *Correctness of Protocols in Distributed Systems.* PhD thesis, Department of Computer Science, Technical University of Denmark, Lyngby, Denmark, December 1993. ID-TR: 1993-131. Also, expanded version in MIT/LCS/TR-589.

[112] Jørgen Søgaard-Andersen, Nancy A. Lynch, and Butler Lampson. Correctness of communication protocols: A case study. Technical Memo MIT/LCS/TR-589, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139, November 1993. Expanded version of [111].

[113] Ekrem Söylemez. Automatic verification of the timing properties of MMT automata. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, February 1994.

[114] Frits Vaandrager and Nancy Lynch. Action transducers and timed automata. In W. R. Cleaveland, editor, *CONCUR'92* (Third International Conference on Concurrency Theory, Stony Brook, NY, USA, August 1992), volume 630 of *Lecture Notes in Computer Science*, pages 436–455. Springer-Verlag, 1992.

[115] Mandana Vaziri. Proving correctness of a controller algorithm for the RAID level 5 system. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, August 1996.

[116] H. B. Weinberg. Correctness of vehicle control systems: A case study. Technical Report MIT/LCS/TR-685, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, February 1996. Masters thesis.

[117] H. B. Weinberg and Nancy Lynch. Correctness of vehicle control systems: A case study. In *17th IEEE Real-Time Systems Symposium*, pages 62–72, Washington, D. C., December 1996. Earlier version appears as [79].

[118] H. B. Weinberg, Nancy Lynch, and Norman Delisle. Verification of automated vehicle protection systems. In R. Alur, T. Henzinger, and E. Sontag, editors, *Hybrid Systems III: Verification and Control* (DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, New Brunswick, New Jersey, October 1995), volume 1066 of *Lecture Notes in Computer Science*, pages 101–113. Springer-Verlag, 1996.

Attachment A

Bibliography of Papers Produced During the Contract Period

[1] S. Abiteboul, G.M. Kuper, H.G. Mairson, A.A. Shvartsman, and M.Y. Vardi. In memoriam: Paris C. Kanellakis, a technical obituary. *ACM Computing Surveys*, March 1996.

[2] Y. Afek, B. Awerbuch, E. Gafni, Y. Mansour, A. Rosen, and N. Shavit. Slide: the key to polynomial end-to-end communication. *Journal of Algorithms*. To appear.

[3] Sudhanshu Aggarwal, Juan Garay, and Amir Herberg. Adaptive video on demand (abstract). In *Proceedings of the Thirteenth Annual ACM Symposium on Principles of Distributed Computing*, August 1994.

[4] Sudhanshu Aggarwal and Shay Kutten. Time optimal self stabilizing spanning tree algorithms. In R.K. Shyamasundar, editor, *13th International Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 761 of *Lecture Notes in Computer Science*, pages 400–410, Bombay, India., December 1993. Springer-Verlag.

[5] Hagit Attiya, Amir Herzberg, and Sergio Rajsbaum. Optimal clock synchronization under different delay assumptions. In *Proceedings of the Twelfth Annual ACM Symposium on the Principles of Distributed Computing*, pages 109–120, Ithaca, NY, August 1993.

[6] H. Attiya, A. Herzberg, and S. Rajsbaum. Optimal clock synchronization under different delay assumptions. Technical Memo MIT/LCS/TM-504, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, April 1994.

[7] H. Attiya and N. Lynch. Time bounds for real-time process control in the presence of timing uncertainty. *Inf. Comput.*, 110(1):183–232, April 1994.

[8] Hagit Attiya, Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Bounds on the time to reach agreement in the presence of timing uncertainty. *Journal of the ACM*, 41(1):122–152, January 1994.

[9] Hagit Attiya and Sergio Rajsbaum. A combinatorial framework for wait-free computability. Technical Report 95/3, Digital Equipment Corporation, Cambridge Research Lab, Cambridge, MA 02139, March 1995.

[10] B. Awerbuch, Y. Azar, and R. Gawlick. Competitive on-line routing and admission control for multicast. Manuscript, 1994.

[11] B. Awerbuch, L. Cowen, and M. Smith. Efficient asynchronous distributed symmetry breaking. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 214–223, May 1994.

[12] B. Awerbuch, R. Gawlick, T. Leighton, and Y. Rabani. On-line admission control and circuit routing for high performance computation and communication. In *Proceedings of the* 35^{th} *Annual Symposium on Foundations of Computer Science*, pages 412–423, Santa Fe, New Mexico, November 1994.

[13] Baruch Awerbuch, Boaz Patt-Shamir, and George Varghese. Bounding the unbounded. In *Proceedings of the 13th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '94)*, pages 776-783, Toronto, Ontario, 1994.

[14] Baruch Awerbuch, Boaz Patt-Shamir, and George Varghese. Self-stabilization by local checking and global reset. In In *Proceedings of the 8th International Workshop on Distributed Algorithms (WDAG'94)*, volume 857 of Lecture Notes in Computer Science, pages 326–339, Terschelling, The Netherlands, September 1994.

[15] Baruch Awerbuch, Boaz Patt-Shamir, and George Varghese. Self-stabilizing end-to-end communication. *Journal of High Speed Networks*, 5(4):365-381, 1996.

[16] Michael S. Branicky, Ekaterina Dolginova, and Nancy Lynch. A toolbox for proving and maintaining hybrid specifications. Presented at *HS'96: Hybrid Systems*, October 12-16, 1996, Cornell University, Ithacs, NY. To be published in proceedings.

[17] James E. Burns and Nancy A. Lynch. Bounds on shared memory for mutual exclusion. *Inf. Comput.*, 107(2):171–184, December 1993.

[18] J.F. Buss, P.C. Kanellakis, P. L. Ragde, and A. Shvartsman. Parallel algorithms with processor failures and delays. *Journal of Algorithms*, 20:45–86, January 1996.

[19] Soma Chaudhuri, Hagit Attiya, Roy Friedman, and Jennifer Welch. Shared memory consistency conditions for non-sequential execution: Definitions and programming strategies. In *5th Annual ACM Symposium on Parallel Algorithms and Architectures*, July 1993. A detailed version appears as Technical Report LPCR 9302, Laboratory for Parallel Computing Research, Department of Computer Science, The Technion, 1993.

[20] Soma Chaudhuri, Rainer Gawlick, and Nancy Lynch. Designing algorithms for distributed systems with partially synchronized clocks. In *Proceedings of the 12th Annual ACM Symposium on the Principles of Distributed Computing*, pages 121–132, Ithaca, New York, USA, August 1993.

[21] Soma Chaudhuri, Maurice Herlihy, Nancy A. Lynch, and Mark R. Tuttle. Tight bounds for k-set agreement. Technical Report Technical Memo CRL00/0, Digital Equipment Corporation, Cambridge Research Lab, September 1993.

[22] Soma Chaudhuri, Maurice Herlihy, Nancy A. Lynch, and Mark R. Tuttle. A tight lower bound for processor coordination. In Donald S. Fussell and Miroslaw Malek, editors, *Responsive Computer Systems: Steps Toward Fault-Tolerant Real-Time Systems*, chapter 1, pages 1–18. Kluwer Academic Publishers, Boston, MA, 1995. Selected papers from the *Second International Workshop on Responsive Computer Systems* Lincoln, New Hampshire, September 28-30, 1993).

[23] Soma Chaudhuri, Maurice Herlihy, Nancy A. Lynch, and Mark R. Tuttle. A tight lower bound for k-set agreement. In *Proceedings of the* 34^{th} *Annual Symposium on Foundations of Computer Science*, pages 206–215, Palo Alto, California, USA, November 1993.

[24] Ranjan Das and Alan Fekete. Modular reasoning about open systems: A case study of distributed commit. In *Proceedings of Seventh International Workshop on Software Specification and Design*, pages 30–39, Los Angeles, CA, December 1993.

[25] Giovanni Della Libera and Nir Shavit. Reactive Diffracting Trees. *Proceedings of the 9th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, Newport, Rhode Island, June 1997. To appear.

[26] Roberto DePrisco, Butler Lampson, and Nancy Lynch. Revisiting the Paxos Algorithm. Manuscript, January 1997.

[27] Harish Devarajan, Alan Fekete, Nancy Lynch, and Liuba Shrira. Correctness proof for a network synchronizer. Technical Memo MIT/LCS/TR-588, Laboratory for Computer Science, Massachusetts Institute of Technology, December 1993.

[28] Danny Dolev and Nir Shavit. Bounded time stamping. *SIAM Journal on Computing*, 26(2):418-455, April 1997.

[29] Ekaterina Dolginova and Nancy Lynch. Safety Verification for Automated Platoon Maneuvers: A Case Study. In Oded Maler, editor *Hybrid and Real-Time Systems* (International Workshop, HART97, Grenoble, France, March 1997), volume 1201 in *Lecture Notes in Computer Science*, pages 154-170, Springer-Verlag, Berlin Heidelberg, 1997.

[30] S. Even and S. Rajsbaum. The use of a synchronizer yields maximum computation rate in distributed networks. *Theory of Computing Systems (formerly Mathematical Systems Theory*, 1997. To appear.

[31] Alan Fekete, David Gupta, Victor Luchangco, Nancy Lynch, and Alex Shvartsman. Eventually-serializable data services. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 300–309, Philadelphia, PA, May 1996.

[32] A. Fekete, N. Lynch, Y. Mansour, and J. Spinelli. The impossibility of implementing reliable communication in the face of crashes. *Journal of the ACM*, 40(5), November 1993.

[33] Alan Fekete and Nancy Lynch and Alex Shvartsman. Specifying and Using a Partitionable Group Communication Service. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA, August 1997. To appear.

[34] Alan Fekete, Nancy Lynch, and William Weihl. Hybrid atomicity for nested transactions. *Theoretical Computer Science B (Logic, semantics and theory of programming)*, 149(1):151–178, September 1995. Special issue of *TCS* devoted to *ICDT '92*.

[35] Alan Fekete, M. Frans Kaashoek, and Nancy Lynch. Implementing sequentially consistent shared objects using broadcast and point-to-point communication. In *Proceedings of the 15th International Conference on Distributed Computing Systems*, pages 439–449, Vancouver, Canada, May/June 1995. IEEE.

[36] Alan Fekete, M. Frans Kaashoek, and Nancy Lynch. Implementing sequentially consistent shared objects using broadcast and point-to-point communication. Technical Memo MIT/LCS/TM-518, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, June 1995.

[37] Alan Fekete, M. Frans Kaashoek, and Nancy Lynch. Implementing Sequentially Consistent Shared Objects Using Broadcast and Point-to-Point Communication. Submitted for journal publication.

[38] F. Fich, M. Herlihy, and N. Shavit. On the complexity of randomized synchronization. In *Proceedings of the* 12^{th} *Annual ACM Symposium on Principles of Distributed Systems*, Ithaca, NY, August 1993.

[39] H. Galeana-Sanchez and S. Rajsbaum. Cycle Pancyclism in Tournaments I. *Graphs and Combinatorics*, 11:233-243, 1995.

[40] H. Galeana–Sanchez and S. Rajsbaum. Cycle Pancyclism in Tournaments II. *Graphs and Combinatorics*, 12:9-16, 1996.

[41] R. Gawlick, C. Kalmanek, and K. Ramakrishnan. On-line routing for permanent virtual circuits. In *Proceedings of IEEE INFOCOM 95: Fourteenth Annual Joint Conference of the IEEE Computer and Communication Societies*, pages 278–288, Boston, Massachusetts, April 1995.

[42] Rainer Gawlick, Charles Kalmanek, and K. G. Ramakrishnan. On-line routing for permanent virtual circuits. *Computer Communications*, 19:235–244, 1996.

[43] R. Gawlick, A. Kamath, S. Plotkin, and K. Ramakrishnan. Routing and admission control of virtual circuits in general topology networks. Submitted for publication, 1995.

[44] Roberto Segala and Rainer Gawlick and Jørgen Søgaard-Andersen and Nancy Lynch. Liveness in Timed and Untimed Systems. Submitted for journal publication.

[45] R. Gawlick, R. Segala, J. Søgaard-Andersen, and N. Lynch. Liveness in timed and untimed systems. Technical Report MIT/LCS/TR-587, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 02139, December 1993.

[46] Rainer Gawlick, Roberto Segala, Jørgen Søgaard-Andersen, and Nancy Lynch. Liveness in timed and untimed systems. In Serge Abiteboul and Eli Shamir, editors, *Proceedings of the 21st International Colloquim, ICALP94*, volume 820 of *Lecture Notes in Computer Science*, pages 166–177, Jerusalem, Israel, July 1994. Springer-Verlag.

[47] Rainer Gawlick, Nancy Lynch, and Nir Shavit. Concurrent time-stamping made simple. Full version submitted for journal publication, 1995.

[48] Michel Goemans, Nancy Lynch, and Isaac Saias. Upper and lower bounds on the number of faults a system can withstand without repairs. In *Fourth International Working Conference on Dependable Computing for Critical Applications*, pages 260–269, San Diego, CA, USA, January 1994.

[49] Kenneth J. Goldman and Nancy Lynch. Quorum consensus in nested transaction systems. *ACM Transactions on Database Systems*, 19(4):537-585, December 1994.

[50] Constance Heitmeyer and Nancy Lynch. The generalized railroad crossing: A case study in formal verification of real-time systems. In *Proceedings of the 13th Real-Time Systems Symposium.*, pages 120-131, San Juan, Puerto Rico, December 1994. IEEE.

[51] Constance Heitmeyer and Nancy Lynch. The generalized railroad crossing: A case study in formal verification of real-time systems. Technical Memo MIT/LCS/TM-511, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, November 1994.

[52] Constance Heitmeyer and Nancy Lynch. Formal verification of real-time systems using timed automata. In Constance Heitmeyer and Dino Mandrioli, editors, *Formal Methods for Real-Time Computing*, Trends in Software, chapter 4, pages 83–106. John Wiley & Sons Ltd, April 1996.

[53] M. Herlihy, B. H. Lim, and N. Shavit. Scalable concurrent counting. *ACM Transactions on Computer Systems*, 13(4):343–364, 1995.

[54] Maurice Herlihy and Sergio Rajsbaum. Set consensus using arbitrary objects. In *Thirteenth Annual ACM Symposium on the Principles of Distributed Computing*, pages 324–333, Los Angeles, CA, August 1994.

[55] Maurice Herlihy and Sergio Rajsbaum. Algebraic spans. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 90–99, Ottawa, Ontario, Canada, August 1995.

[56] Maurice Herlihy, Nir Shavit, and Orli Waarts. Linearizable counting networks. *Distributed Computing*, (9):193–203, 1996.

[57] M. P. Herlihy and N. Shavit. The topological structure of asynchronous computability. To appear as a Brown University TR. Also, submitted for publication.

[58] Gunnar Hoest and Nir Shavit. Towards a Topological Characterization of Asynchronous Complexity. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA, August 1997. To appear.

[59] P.C. Kanellakis, D. Michailidis, and A. Shvartsman. Controlling memory access in efficient fault-tolerant parallel algorithms. *Nordic Journal of Computing*, 2:146–180, 1995.

[60] Paris C. Kanellakis and Alex A. Shvartsman. *A theory of fault-tolerant parallel computation.* Kluwer Academic Publishers, Boston, MA 1997.

[61] Jon Kleinberg, Hagit Attiya, and Nancy Lynch. Trade-offs between message delivery and quiesce times in connection management protocols. In *Proceedings of ISTCS 1995: The Third Israel Symposium on Theory of Computing and Systems*, pages 258–267, Tel-Aviv, Israel, January 1995. IEEE.

[62] Butler W. Lampson, Nancy A. Lynch, and Jørgen F. Søgaard-Andersen. Correctness of at-most-once message delivery protocols. In Richard L. Tenney, Paul D. Amer, and M. Ümit Uyar, editors, *Formal Description Techniques, VI* (Proceedings of the IFIP TC6/WG6.1 Sixth International Conference on Formal Description Techniques — FORTE'93, Boston, MA, October 1993), pages 385–400. North-Holland, 1994.

[63] Gunter Leeb and Nancy Lynch. Proving Safety Properties of the Steam Boiler Controller: Formal Methods for Industrial Applications: A Case Study, 1996. To appear in *Lecture Notes in Computer Science*, Springer-Verlag series. Earlier version presented at the *Methods for Semantics and Specification* (International Conference and Research Center for Computer Science, Schloss, Dagstuhl, Germany, June 1995), as "Using Timed Automata for the Steam Boiler Controller Problem."

[64] Victor Luchangco, Ekrem Söylemez, Stephen Garland, and Nancy Lynch. Verifying timing properties of concurrent algorithms. In Dieter Hogrefe and Stefan Leue, editors, *Formal Description Techniques VII: Proceedings of the 7th IFIP WG6.1 International Conference on Formal Description Techniques* (FORTE'94, Berne, Switzerland, October 1994), pages 259–273. Chapman and Hall, 1995.

[65] John Lygeros and Nancy Lynch. Formal verification of the TCAS conflict resolution algorithms. In *1997 Conference on Decision and Control*, San Diego, CA, December 1997. To appear. Extended abstract.

[66] Nancy Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, March 1996.

[67] Nancy Lynch. Modelling and verification of automated transit systems, using timed automata, invariants and simulations. In R. Alur, T. Henzinger, and E. Sontag, editors, *Hybrid Systems III: Verification and Control* (DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, New Brunswick, New Jersey, October 1995), volume 1066 of *Lecture Notes in Computer Science*, pages 449–463. Springer-Verlag, 1996.

[68] Nancy Lynch. Modelling and verification of automated transit systems, using timed automata, invariants and simulations. Technical Memo MIT/LCS/TM-545, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, December 1995.

[69] Nancy Lynch. Simulation techniques for proving properties of real-time systems. In *A Decade of Concurrency: Reflections and Perspectives*, Lecture Notes in Computer Science, pages 375–424, REX School/Symposium, Noordwijkerhout, the Netherlands, June 1993. Springer-Verlag.

[70] Nancy Lynch. Simulation techniques for proving properties of real-time systems. Technical Memo MIT/LCS/TM-494, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, November 1993.

[71] Nancy Lynch. Atomic transactions for multiprocessor programming: A formal approach. In Guy E. Blelloch, K. Mani Chandy, and Suresh Jagannathan, editors, *Specification of Parallel Algorithms: DIMACS Workshop*, volume 18 of *Discrete Mathematics and Theoretical Computer Science*, pages 125–142, Princeton, NJ, May 1994. Amercian Mathematical Society.

[72] Nancy Lynch. Simulation techniques for proving properties of real-time systems. In Sang H. Son, editor, *Advances in Real-Time Systems*, chapter 13, pages 299–332. Prentice Hall, Inc., Englewood Cliffs, NJ, 1995.

[73] Nancy Lynch. Proving performance properties (even probabilistic ones). In Dieter Hogrefe and Stefan Leue, editors, *Formal Description Techniques VII: Proceedings of the 7th IFIP WG6.1 International Conference on Formal Description Techniques (FORTE'94)*, pages 3–20. Chapman and Hall, 1995. Invited talk.

[74] Nancy Lynch, Michael Merritt, William Weihl, and Alan Fekete. *Atomic Transactions.* Morgan Kaufmann Publishers, 1994.

[75] Nancy Lynch, Isaac Saias, and Roberto Segala. Proving time bounds for randomized distributed algorithms. In *Proceedings of the Thirteenth Annual ACM Symposium on the Principles of Distributed Computing*, pages 314–323, Los Angeles, CA, August 1994.

[76] Nancy Lynch and Sergio Rajsbaum. On the Borowsky-Gafni simulation algorithm. In *Proceedings of the Fourth ISTCS: Israel Symposium on Theory of Computing and Systems*, pages 4–15, Jerusalem, Israel, June 1996. IEEE Computer Society. Also, short version appears in *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, Philadelphia, PA, page 57, May 1996.

[77] N. Lynch and R. Segala. A comparison between simulation techniques and algebraic techniques for verifying concurrent systems. In *NAPAW Proceedings of the North American Process Algebra Workshop*, Department of Computer Science, Cornell University, Ithaca, NY, August 1993. TR 93-1369.

[78] N. Lynch and R. Segala. A comparison of simulation techniques and algebraic techniques for verifying concurrent systems. *Formal Aspects of Computing*, 7(3):231–265, 1995.

[79] Nancy Lynch and Roberto Segala. A comparison of simulation techniques and algebraic techniques for verifying concurrent systems. Technical Memo MIT/LCS/TM-499, Laboratory for Computer Science, Massachusetts Institute Technology, December 1993.

[80] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H. B. Weinberg. Hybrid I/O automata. In R. Alur, T. Henzinger, and E. Sontag, editors, *Hybrid Systems III: Verification and Control* (DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, New Brunswick, New Jersey, October 1995), volume 1066 of *Lecture Notes in Computer Science*, pages 496–510. Springer-Verlag, 1996.

[81] Nancy Lynch, Roberto Segala, Frits Vaandrager, and H. B. Weinberg. Hybrid I/O automata. Technical Memo MIT/LCS/TM-544, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, December 1995.

[82] Nancy Lynch and Alex Shvartsman. Robust Emulation of Shared Memory Using Dynamic Quorum-acknowledged broadcasts, *Twenty-Seventh Annual International Symposium on Fault-Tolerant Computing* (FTCS'97), Seattle, Washington, USA, June 1997. To appear.

[83] Nancy Lynch, Nir Shavit, Alex Shvartsman, and Dan Touitou. Counting networks are practically linearizable. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 280–289, Philadelphia, PA, May 1996.

[84] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part II: Timing-based systems. *Information and Computation*, 128(1):1–25, July 1996.

[85] Nancy A. Lynch and Frits W. Vaandrager. Action transducers and timed automata. *Formal Aspects of Computing.* 8(5):499-538, 1996.

[86] Nancy Lynch and Frits Vaandrager. Action transducers and timed automata. Technical Memo MIT/LCS/TM-480.c, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, July 1995.

[87] Nancy Lynch and Frits Vaandrager. Action transducers and timed automata. Technical Memo MIT/LCS/TM-480.b, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, October 1994.

[88] Nancy Lynch and Frits Vaandrager. Forward and backward simulations – Part I: Untimed systems. Technical Memo MIT/LCS/TM-486.b, Laboratory for Computer Science, Massachusetts Institute Technology, Cambridge, MA, 02139, August 1994. (Replaces MIT/LCS/TM-486, April 1993).

[89] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part I: Untimed systems. *Information and Computation*, 121(2):214–233, September 1995.

[90] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part II: Timing-based systems. Technical Memo MIT/LCS/TM-487.c, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, April 1995.

[91] Nancy Lynch and Frits Vaandrager. Forward and backward simulations — Part II: Timing-based systems. *Information and Computation*, 128(1):1–25, July 1996.

[92] Nancy Lynch and H. B. Weinberg. Proving correctness of a vehicle maneuver: Deceleration. In *Second European Workshop on Real-Time and Hybrid Systems*, pages 196–203, Grenoble, France, May/June 1995. Later version appears as [122].

[93] Yishay Mansour and Boaz Patt-Shamir. Many-to-one packet routing on grids. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on the Theory of Computing*, pages 258–267, Las Vegas, Nevada, May/June 1995.

[94] R. De Nicola and R. Segala. A process algebraic view of I/O automata. *Theoretical Computer Science*, 138:391–423, March 1995. Also, Rapporto Tecnico N. SI-92/05, Universitá "La Sapienza", Rome.

[95] Boaz Patt-Shamir and Sergio Rajsbaum. A theory of clock synchronization. In *Proceedings of the* 26^{th} *Annual ACM Symposium on Theory of Computing*, pages 810–819, Montreal, Canada, May 1994. Journal version in progress.

[96] Tsvetomir P. Petrov, Anna Pogosyants, Stephen J. Garland, Victor Luchangco, and Nancy A. Lynch. Computer-assisted verification of an algorithm for concurrent timestamps. In Reinhard Gotzhein and Jan Bredereke, editors, *Formal Description Techniques IX: Theory, Applications, and Tools* FORTE/PSTV'96: Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification, Kaiserslautern, Germany, October 1996, pages 29–44. Chapman & Hall, 1996.

[97] Anna Pogosyants and Roberto Segala. Formal verification of timed properties of randomized distributed algorithms. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 174–183, Ottawa, Ontario, Canada, August 1995.

[98] Anna Pogosyants, Roberto Segala, and Nancy Lynch. Verification of the Randomized Consensus Algorithm of Aspnes and Herlihy: A Case Study. Manuscript. Also, submitted for journal publication and to be MIT/LCS/TM-555.

[99] Yaron Riany, Nir Shavit, and Dan Touitou. Towards a practical snapshot algorithm. In *Proceedings of ISTCS 1995: The Third Israel Symposium on the Theory of Computing Systems*, pages 121–129, Tel-Aviv, Israel, January 1995. IEEE.

[100] S. Rajsbaum. Upper and lower bounds for stochastic marked graphs. *Inf. Process. Lett.*, 49:291–295, 1994. Preliminary version appeared as "Stochastic Marked Graphs," in *Proceedings of the 4th. International Workshop on Petri Nets and Performance Models,* Melbourne, Australia, diciembre 3–5, 1991, pp. 95–101.

[101] Roberto Segala. Quiescence, fairness, testing and the notion of implementation. In Eike Best, editor, *Proceedings of CONCUR 93: 4th International Conference on Concurrency Theory*, volume 715 of *Lecture Notes in Computer Science*, pages 324–338, Hildesheim, Germany, August 1993. Springer-Verlag.

[102] Roberto Segala. Quiescence, fairness, testing and the notion of implementation. *Information and Computation.* To appear, 1997.

[103] Roberto Segala and Nancy Lynch. Probabilistic simulations for probabilistic processes. In Bengt Jonsson and Joachim Parrow, editors, *CONCUR'94: Concurrency Theory* (5th International Conference, Uppsala, Sweden, August 1994), volume 836 of *Lecture Notes in Computer Science*, pages 481–496. Springer-Verlag, 1994. A revised version in [104]

[104] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250-273, August 1995. Special issue on selected papers from CONCUR94.

[105] Roberto Segala. A compositional trace-based semantics for probabilistic automata. In Insup Lee and Scott A. Smolka, editors, *CONCUR 95: Concurrency Theory* (6th International Conference, Philadelphia, Pennsylvania, August 1995), volume 962 of *Lecture Notes in Computer Science*, pages 234–248. Springer-Verlag, 1995.

[106] Nir Shavit and Dan Touitou. Elimination trees and the construction of pools and stacks. In *SPAA'95: 7th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 54–63, Santa Barbara, California, July 1995.

[107] Nir Shavit and Dan Touitou. Elimination trees and the construction of pools and stacks. *Theory of Computing Systems (formerly Mathematical Systems Theory)*, 1997. To appear.

[108] Nir Shavit and Dan Touitou. Software transactional memory. In *Proceedings of the Fourteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 204–213, Ottawa, Ontario, Canada, August 1995.

[109] Nir Shavit and Dan Touitou. Software transactional memory. Technical Memo MIT/LCS/TM-675, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, 1996.

[110] Nir Shavit and Dan Touitou. Software Transactional Memory. *Distributed Computing.* 10(2):99-116, February 1997.

[111] N. Shavit, E. Upfal, and A. Zemach. A Steady State analysis of Diffracting Trees. In *Proceedings of the 8th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 33–41, Padua, Italy, June 1996.

[112] N. Shavit and E. Upfal, and A. Zemach. A Steady State Analysis of Diffracting trees. *Theory of Computing Systems (formerly Mathematical Systems Theory).* Special Issue. To appear.

[113] N. Shavit, E. Upfal and A. Zemach. A Wait-Free Sorting Algorithm. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA, August 1997. To appear.

[114] Nir Shavit and Asaph Zemach. Diffracting trees. *ACM Transactions on Computer Systems.* 14(4):385-428, November 1996.

[115] Nir Shavit and Asaph Zemach. Diffracting trees. In *Proceedings of the Sixth Annual Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 167–176, Cape May, New Jersey, June 1994.

[116] Alex A. Shvartsman and C. Strutt. A framework for distributed object management and generic applications. Manuscript, 1995.

[117] Mark Smith. Formal verification of communication protocols. In Reinhard Gotzhein and Jan Bredereke, editors, *Formal Description Techniques IX: Theory, Applications, and Tools* FORTE/PSTV'96: Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification, Kaiserslautern, Germany, October 1996, pages 129–144. Chapman & Hall, 1996.

[118] Mark Smith. Formal Verification of Communication Protocols. *6th Annual MIT Student Workshop on Computing Technology*, Salem, MA. August 15, 1996. Extended abstract.

[119] Jørgen Søgaard-Andersen, Nancy A. Lynch, and Butler Lampson. Correctness of communication protocols: A case study. Technical Memo MIT/LCS/TR-589, Laboratory for Computer Science, Massachusetts Institute Technology, Cambridge, MA, 02139, November 1993. Also, PhD Thesis, Technical University of Denmark, Lyngby, Denmark, December 1993.

[120] George Varghese. Self-stabilization by local checking and correction. Technical Report MIT/LCS/TR-583, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, October 1993.

[121] George Varghese and Nancy A. Lynch. A tradeoff between safety and liveness for randomized coordinated attack protocols. *Information and Computation*, 128(1):57-71, July 1996.

[122] H. B. Weinberg and Nancy Lynch. Correctness of vehicle control systems: A case study. In *17th IEEE Real-Time Systems Symposium*, Washington, D. C., pages 62-72, December 1996.

[123] H.B. Weinberg, Nancy Lynch, and Norman Delisle. Verification of automated vehicle protection systems. In R. Alur, T. Henzinger, and E. Sontag, editors, Hybrid Systems III: Verification and Control (DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, New Brunswick, New Jersey, October 1995), volume 1066 of Lecture Notes in Computer Science, pages 101-113. Springer-Verlag, 1996.

Papers in progress

B. Chlebus, R. De Prisco, and A. Shvartsman. "Work in a Message-passing Environment Prone to Processor Failures and Restarts."

Matteo Frigo and Victor Luchangco. "Computation-Centric Memory Models."

Stephen Garland and Mandana Vaziri. "IOA: A Formal Language for I/O Automata."

Henrik Jensen. "Abstraction Methods for Model Checking in the I/O Automata Framework."

Jon Kleinberg, Hagit Attiya, and Nancy Lynch. "Trade-offs between Message Delivery and Quiesce Times in Connection Management Protocols." Journal version.

Victor Luchangco. "Precedence-Based Memory Models."

John Lygeros and Nancy Lynch. "Conditions for Safe Platoon Deceleration."

Nancy Lynch, Roberto Segala, Frits Vaandrager, and H. B. Weinberg. "Hybrid I/O Automata." Journal version.

Nancy Lynch, Nir Shavit, Alex Shvartsman, and Dan Touitou. "Timing Conditions for Linearizability in Counting Networks." Journal version.

Nancy Lynch and Sergio Rajsbaum. "On the Borowsky-Gafni Simulation Algorithm." Journal version.

Shavit and Shvartsman. Journal version of "Counting Networks are Practically Linearizable" coauthored with Dan Touitou and Nancy Lynch. In progress.

Mark Smith and Nancy Lynch. "The Impossibility of At-Most-Once Fast Message Delivery."

Attachment B

Phd Theses Completed

[1] Rainer Gawlick. *ATM Routing: Theory and Practice.* PhD thesis, MIT Dept. of Electrical Engineering and Computer Science, June 1995. Also, MIT/LCS/TR-679.

[2] Boaz Patt-Shamir. *A Theory of Clock Synchronization.* PhD thesis, MIT Electrical Engineering and Computer Science, October 1994. Also, MIT/LCS/TR-680.

[3] Isaac Saias. *Randomness versus Non-Determinism in Distributed Computing.* PhD thesis, Massachusetts Institute of Technology, Mathematics Department, Cambridge, MA, October 1994. Also, MIT/LCS/TR-651.

[4] Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, MIT Dept. of Electrical Engineering and Computer Science, May 1995. Also, MIT/LCS/TR-676.

[5] Jørgen Søgaard-Andersen. *Correctness of Protocols in Distributed Systems.* PhD thesis, Technical University of Denmark, Lyngby, Denmark, December 1993. ID-TR: 1993-131. Also, MIT/LCS/TR-589.

Masters Theses Completed

[6] Sudhanshu Aggarwal. Time optimal self-stabilizing spanning tree algorithms. Master's thesis, MIT Electrical Engineering and Computer Science, May 1994. Also, MIT/LCS/TR-632.

[7] Giovanni Della-Libera. Reactive elimination trees. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, July, 1996.

[8] Roberto De Prisco. Revisiting the Paxos algorithm. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, June 1997. Also, to be MIT/LCS/TR-717.

[9] Victor Luchangco. Using simulation techniques to prove timing properties. Master's thesis, MIT Electrical Engineering and Computer Science, Cambridge, MA 02139, June 1995.

[10] Ekrem Söylemez. Automatic verification of the timing properties of MMT automata. Master's thesis, MIT Dept. of Electrical Engineering and Computer Science, February 1994.

[11] Mandana Vaziri. Proving Correctness of a Controller Algorithm for the RAID Level 5 System. Masters thesis, Dept. of Electrical Engineering and Computer Science, Cambridge, MA 02139, August 1996.

[12] H. B. Weinberg. Correctness of Vehicle Control Systems: A Case Study. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, February 1996. Also, MIT/LCS/TR-685.

PhD Theses In Progress

[13] Victor Luchangco. *Building Blocks for Distributed Computing Applications.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, 1996. In progress.

[14] Mark Smith. *Formal Verification of TCP and T/TCP.* PhD thesis, Dept. of Electrical Engineering and Computer Science, Cambridge, MA 02139, 1997. In progress.

[15] Mandana Vaziri. PhD thesis (Untitled). Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139. In progress.

Masters Theses In Progress

[16] Oleg Cheiner. Implementation and Evaluation of an Eventually-Serializable Data Service. Masters thesis, Dept. of Electrical Engineering and Computer Science, Cambridge, MA 02139, in progress.

[17] Gunnar Hoest. Towards a Topological Characterization of Complexity in Asynchronous, Distributed Systems. Masters thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139. In progress.

[18] Roger Khazan. Group Communication as a Base for a Load-Balancing, Replicated Data Service. Masters thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139. In progress.

[19] Carolos Livadas. Verification of Automated Vehicle Protection Systems. Masters thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, 1996. In progress.

In Memory of

[20] Anya Pogosyants. *Formal Verification of Randomized Distributed Systems.* PhD thesis, Dept. of Electrical Engineering and Computer Science, Cambridge, MA 02139, 1995. (Never completed due to her death).